L16: Entry 8 of 12 File: USPT Jun 18, 2002

DOCUMENT-IDENTIFIER: US 6408310 B1
TITLE: System and method for expediting transfer of sectioned audit files from a primary host to a secondary host

Abstract Text (1):
In order to correlate or synchronize the database of a source primary host with the backup database of a remote secondary host, the present method operates to speed up the transfer of audit blocks from primary host to secondary host when an out-of-sync signal indicates the secondary host audit blocks are not on a par with the primary host audit blocks. A Catchup process utilizes multiple buffers of audit blocks at the primary host which can then be asynchronously transferred, by a sequence of Catchup tasks, on a parallel concurrent operation, over to physical files of said secondary host.

Parent Case Text (2):
This application is related to co-pending applications, U.S. Ser. No. 09/374,351 entitled: "Enhanced System and Method for Management of System Database Utilities"; U.S. Ser. No. 09/415,273 entitled "System and Method for Asynchronously Receiving Multiple Packets of Audit Data From a Source Database Host in a Resynchronization Mode and Asynchronously Writing the Data to a Target Host"; and U.S. Ser. No. 09/415,333 entitled "Tracker Sensing Method for Regulating Synchronization of Audit Files Between Primary and Secondary Hosts", which are incorporated herein by reference.

Brief Summary Text (2):
This disclosure relates to network communications where logical audit blocks are created at a source host and transferred to a remote host where the audit trail is used to create and maintain a continuously synchronized remote database backup.

Brief Summary Text (4):
A database such as the Unisys Data Management System II, Extended, is a centralized collection of data placed into one or more files. Multiple application programs can access this data concurrently. Consequently, redundant files are not required for each individual application. Application programs running in batch, time sharing, and reload job entry environments can all access the database concurrently. A database of the present configuration consists of the following major components:

Brief Summary Text (10):
A data set, a set, or a subset, that is not an item of another set is termed disjoint. Structures need not be disjoint, that is to say a hierarchy can exist between the various data sets, sets, and subsets. A data set, a set, or a subset, that is an item in another data set, is said to be embedded. When a database contains embedded structures, a hierarchical file structure results.

Brief Summary Text (12):
A set is a structure that allows access to all records of a data set in some logical sequence. The set contains one entry for each record in the data set. Each set entry is an index that locates a data set record. If key items are specified for the set, records in the data set are accessed based upon these keys. Otherwise the records are accessed sequentially. Multiple sets can be declared for a single

data set, thereby enabling the data in a data set to be accessed in several different sequences. A subset is similar to a set. Unlike a set, a subset need only refer to selected records in the data set. A data item is a field in a database record used to contain an individual piece of information.

Brief Summary Text (13):
Data items that are not a part of any data set are then called global data items. Global data items generally consist of information such as control totals, hash totals, and populations, which apply to the entire database. All global data items are stored in a single record.

Brief Summary Text (14):
The audit trail is a record of changes made to the database. The audit trail is used to recover automatically the database following a hardware or software failure. The audit trail specification clause describes the physical attributes of the audit trail.

Brief Summary Text (15):
The audit trail, as mentioned, consists of a record of changes to the database. It is only created for audited databases and is used in the various forms of database recovery.

Brief Summary Text (17):
All audited databases must include a "restart" data set definition. There is a specialized syntax for specifying the audit trail attributes. These involve area size, area length, block size, buffers, checksum, and sections in addition to whether disk or tape is involved and types of tape being used.

Brief Summary Text (25):
A Remote Database Backup or RDB is a database recovery system which can be a key component of a disaster recovery plan since it minimizes the amount of time needed to recover from a loss database access. The RDB system also minimizes the loss of productivity, minimizes the loss of revenue and minimizes the loss of business, which could occur because of interruptions in the ability to access one's database. The RDB works in conjunction with the Data Management System II (DMSII) databases plus Structured Query Language Database (SQLDB), the Semantic Information Manager (SIM) database, and the Logic and Information Network Compiler II (LINCII) databases.

Brief Summary Text (26):
The components of the RDB system consist of a database and also a copy of the database. One database is update capable and the other database can be used only for inquiry purposes. The update-capable database is called the primary database. The host on which this database resides is called the primary host. The "current on-line" remote database copy, which is called the secondary database, is "inquiry-capable" only. The host on which this database resides is called the secondary host. The configuration of the primary and the secondary databases on their separate hosts is called the RDB System. A single host can participate in multiple RDB systems.

Brief Summary Text (27):
The RDB or remote database backup system enables users to maintain a current on-line inquiry-only copy of a database on an enterprise server, which is separate from the enterprise server on which the update-capable database resides. The host locations can be at the same site or at two geographically distant sites. The remote database backup keeps the database copy up-to-date by applying the audit images from the audited database to the database copy. There is a choice of four audit transmission modes which enables one to choose the means of audit transfer between hosts.

Brief Summary Text (28):
In the RDB system, the term "primary" and the term "secondary" will indicate the impended function of each copy of the database and the host on which it resides.

Brief Summary Text (29):
The primary database has the function for database inquiry and update, while the secondary database has the functionality useful for database inquiry only.

Brief Summary Text (30):
The secondary database cannot be updated by any application programs and the secondary database is modified only by the application of audit images of transactions performed on the primary database.

Brief Summary Text (31):
Since one complete RDB system is made of one database, and includes the secondary database which resides on another host, that is to say the primary database on one host plus one copy of that database.

Brief Summary Text (32):
A host is the system on which a primary or a secondary database resides. A host can function as a primary host in one RDB system and then also concurrently function as a secondary host for another RDB system. Additionally, one host can function as a secondary host (or a primary host) for multiple RDB systems.

Brief Summary Text (33):
When a RDB system is first initialized for a database, then by default, the primary host is the host upon which the database resides. The other host which is defined for that database is designated as a secondary host and it remains a secondary host until a takeover is performed or until the RDB capability is disabled. Both the primary and secondary hosts must have sufficient resources to support the RDB system and its application environment.

Brief Summary Text (34):
As an illustration, it can be seen how the primary database on a system, which is called Host One and the secondary database is applied on a system called Host Two can work together in response to or in anticipation of an interruption on the primary host. In this example, the application normally runs against the primary database in Host One with the RDB transferring audit images to the secondary database. Under normal operation, which is when the audit images are transferred from the primary database to the secondary database without loss of data during transmission due to network or system failure, the example described above works well. However, in the condition that a network or system failure results in the loss of data during transmission from the primary database to the secondary then the secondary database is said to be out of synchronization with the primary database. Hence there is need of a mechanism by which the lost data can be re-transmitted so that the secondary database can be re-synchronized with the primary one.

Brief Summary Text (35):
The object of the present invention is to expedite and speed-up the transfer of audit blocks from buffers in the primary host for placement into segregated blocks in the secondary host. These segregated blocks of secondary audit data can then be asynchronously moved by a sequence of "Catchup" processes working concurrently in parallel to deposit the audit data onto the secondary backup database.

Brief Summary Text (36):
In order to accomplish this object, there is utilized a logical resynchronization process, which hereinafter is referred to as Catchup, which according to the present system consists of multiple backup database system at the remote host. Initially, the backup system, via a Tracker sensing mechanism, recognizes that a

resynchronization process is required and then, from its shared database library task (RDB Support Library), initiates a single physical Catchup task for each physical audit file partition in a parallel transfer operation.

Brief Summary Text (37):
AUDIT TRAIL SYNCHRONIZATION: It is of some importance to decide on what is called audit level synchronization that is desired for the remote database backup system. This involves the question of "how closely must the backup database match its source database? Or to express it in another fashion, how closely synchronized should the secondary database audit trail be a replicate of the primary database audit trail?"

Brief Summary Text (38):
MODES OF AUDIT TRANSMISSION: The remote database backup (RDB) system provides four specific audit transmission modes that enable the user to regulate whether the transmission of the audit images is to be automatic or manual; whether the transmission of audit images is to be done as individual audit blocks or entirely whole audit files; whether the transmission of audit images can be interrupted, that is to say, suspended or not; and what is to be the degree of audit trail synchronization between the primary host and the secondary host. The focus of the present invention involves the use of one mode designated as the ABW or Audit Block Write mode.

Brief Summary Text (39):
AUDIT BLOCK WRITE (ABW): The secondary audit trail is to be constantly and automatically kept synchronized with the primary database audit trail on a block-by-block basis. The ABW mode enables this type of close synchronization level to occur by (i) handling interruptions to audit transmissions through one of two error handling options; or (ii) initiating a Catchup process for the audit block transfer whenever the usual synchronization level is disrupted. This invention is devoted to the Catchup process.

Brief Summary Text (42):
A method and system for enhancing the rate of transfer speed of audit blocks from a primary host to a secondary host in order to establish a secondary host database which will be in synchronization, i.e., accurately duplicate, the audit file data in the source database of a primary host.

Brief Summary Text (43):
When the network connections between a primary host source and a remote secondary target host are interrupted, disabled or have transmission delays, then the secondary host will be out-of-sync with the source audit file data in said primary host. Ordinary transfer rates for sending audit file data from primary to secondary host would be inadequate to develop data file synchronization between primary and secondary hosts.

Brief Summary Text (44):
Thus, when a Tracker sensing means indicates the lack of synchronism, a Catchup program is initiated which then utilizes audit blocks of sectioned audit files for placement in multiple segregated buffers which are transferred to segregated audit blocks in said secondary host. These segregated audit blocks are each handled asynchronously by a sequence of Catchup processes, working concurrently in parallel which operate to expedite the placement of the audit blocks onto the remote secondary database.

Brief Summary Text (45):
By asynchronously receiving multiple logical audit blocks and asynchronously writing them to multiple physical files, the audit trails become synchronized quicker than if said process were performed in a serial mode of operation.

Drawing Description Text (5):
FIG. 4 is a flow chart showing the steps involved in the audit trail transport process between a primary and secondary database with the use of the Catchup process for expedition of synchronization;

Drawing Description Text (9):
FIG. 8 is a flow chart illustrating the Catchup Server Task for reading audit blocks and writing them asynchronously to the remote host until the audit trails are synchronized.

Detailed Description Text (2):
ACR: Abbreviation for Accessroutines, the software component of the DMSII product that is primarily responsible for the accessing (creating, modifying and deleting) of data in a DMSII database and auditing all changes to the database.

Detailed Description Text (4):
AUDIT DATA: For DMSII databases, data that records every change to a predefined database.

Detailed Description Text (5):
AUDIT FILE: For DMSII databases, a file produced by the Accessroutines that contains various control information, including before and after images of records resulting from changes to the database.

Detailed Description Text (6):
AUDIT FILE VS. AUDIT BLOCK: For DMSII databases, the audit file represents one or more physical files that contain audit blocks that are stored sequentially.

Detailed Description Text (7):
AUDIT FILE SWITCH: For DMSII databases, the logical time when one audit file is complete and a new one is started.

Detailed Description Text (8):
AUDIT IMAGES: For DMSII databases, structured package of data representing change to the database that are stored sequentially into the audit trail.

Detailed Description Text (9):
AUDIT SOFTWARE: These are specialized programs to perform a variety of auditing functions, such as sampling databases or possibly generating confirmation letters to customers. It can be used to highlight certain exceptions to categories of data and alert the user to possible errors. Audit software may often include a non-procedural language that lets the auditor-user describe the computer and the data environment without need for detailed programming.

Detailed Description Text (10):
AUDIT TRAIL: This is a record of transactions in an information system that provides verification of the activity of the system. The simplest audit trail is a transaction itself. For example, if an employees salary is increased, the changed transaction will include the date, the amount of the raise, and the name of the authorizing manager. It is possible to create a more elaborate audit trail when the system is being verified for accuracy. For example, samples of processing results can be recorded at various stages. Item counts and hash totals can be used to verify that all input has been processed through the system. For DMSII databases, the sequence of audit files that are created and span the life of the database.

Detailed Description Text (12):
CATCHUP TASK: In an RDB system, a physical process that runs at a remote host, reads audit data from a port file connected to a source database, and writes the data to a physical audit file.

Detailed Description Text (14):
FASTER AUDIT GENERATION: For DMSII databases, a rate of audit generation that can be achieved by using sectioned audit and multiple processors.

Detailed Description Text (15):
FILE: A collection of bytes which is stored as an individual entity. For example, all data on disk is stored as a file with an assigned file name that is unique within the directory it resides in. To the computer, file is only nothing more than a series of bytes. The structure of a file is known to the software that manipulates it. For example, database files are made up of a series of records. Word processing files (also called documents) contain a continuous flow of text.

Detailed Description Text (17):
FILE FORMAT: This is the structure of a file. There are hundreds of proprietary formats for a database, for word processing, and for graphics files.

Detailed Description Text (19):
FILE MANAGER: (i) This is software that manages data file and Is not to be confused with a database manager. The file managers provide the ability to create, enter, change, query, and produce reports on one single file at a time. There is no relational capability and do not involve a programming language. (ii) Often used for software used to manage files on a disk. It provides functions to delete, copy, remove, rename, and view files as well as to create and manage directories.

Detailed Description Text (21):
FILE SERVER: This is a high speed computer in the local area network (LAN) that stores the programs and the data files shared by users of the network. Sometimes it's called a network server and it acts like a remote disk drive.

Detailed Description Text (22):
LINCII DATABASE: A database generated by the LINC system software; may be a DMSII database.

Detailed Description Text (23):
LOGIC & INFORMATION NETWORK COMPILER (LINC): A software development tool that may be used to generate a DMSII database and any number of applications to access the database.

Detailed Description Text (24):
LOGICAL AUDIT BLOCK: For DMSII databases, a structured package containing potentially many Audit Records (in the extreme situation, it is also possible that a single Audit Block could only contain a partial audit Record).

Detailed Description Text (25):
LOGICAL AUDIT FILE; For DMSII databases, the sequential storage of Audit Blocks that contain Audit Records. One Logical Audit File may contain 1 or more Physical Audit Files (Sections or Partitions). The sequence of Audit Blocks is spread, round robin fashion, among the Audit Sections.

Detailed Description Text (27):
NON-PARTITIONED AUDIT FILE: In a DMSII system, an audit file that has one section or partition. Equally, an audit file that contains one physical file.

Detailed Description Text (28):
NON-SECTIONED AUDIT FILES: Same as NON-PARTITIONED AUDIT FILES.

Detailed Description Text (29):
NORMAL AUDIT TRANSFER: In an RDB system, the uninterrupted transfer of audit data from a source database host to a remote host while the source database is being updated.

Detailed Description Text (30):
ORIGINAL AUDIT TRAIL: In an RDB system, the audit trial of the source database.

Detailed Description Text (31):
PACKET (OF AUDIT DATA): For DMSII databases, a collection of one or more audit blocks.

Detailed Description Text (33):
PARTITIONED AUDIT FILE: For DMSII databases, a logical audit file that is partitioned into a predefined number of physical files.

Detailed Description Text (34):
PERIODIC SYNCHRONIZATION: In an RDB system, audit synchronization that takes place only when complete audit files become available for transfer to a remote host (i.e., following an audit file switch).

Detailed Description Text (35):
PHYSICAL AUDIT FILE: A physical file containing Audit Blocks. May be 1 of many sections of a Logical Audit File.

Detailed Description Text (36):
PORT FILE NETWORK COMMUTATION: In an RDB system, the method of messaging and data transfer between a source database system and a remote backup system.

Detailed Description Text (37):
REMOTE DATABASE BACKUP: A disaster recovery capability for DMSII-based databases that enables the replication of an audit (primary) database on a secondary host. The replicated (secondary) database is kept up-to-date with the primary database through the application of audits from the primary database. When the primary database becomes unavailable, the secondary database can take over the role of the primary database.

Detailed Description Text (38):
REMOTE HOST: In an RDB system, the host that contains the duplicate copy of the source database. Also know as the Secondary Host.

Detailed Description Text (39):
RDB SYSTEM; (Remote Database Backup); This is a Unisys Corporation system for backup of a database and is referenced by a Unisys Publication Item 8600-2052-304 dated December, 1998, entitled "Remote Database Backup--Planning and Operations Guide."

Detailed Description Text (40):
RDBSUPPORT LIBRARY: In an RDB system, the library that is accessed by the shared task, database utilities, and additional applications responsible for configuring an RDB system. The library is also a running process responsible for initiating local and remote tasks through port file communication.

Detailed Description Text (42):
RESYNCHRONIZATION MODE: Under the ABW audit file transmission mode of an RDB database, the process of bringing the audit trail of the secondary database back into the closest possible synchronization with the audit trial of the primary database. Also see Catchup.

Detailed Description Text (43):
SECTIONAL AUDIT FILES: Same as PARTITIONED AUDIT FILES.

Detailed Description Text (44):
SEMANTIC INFORMATION MANAGER (SIM): A database management system that simplifies

the task of modeling your application environment based on the semantic data model.

Detailed Description Text (45):
SERVER TASK: In an RDB system, a task that is connected to a remote host for messaging and data transfer.

Detailed Description Text (46):
SHARED DATABASE TASK: For DMSII databases, the running process accessed by all database applications to read and write data to the database and audit trail.

Detailed Description Text (47):
SIM-DATABASE: A DMSII database defined by SIM.

Detailed Description Text (48):
SOURCE DATABASE HOST: In an RDB system, the host that contains the primary copy of the database.

Detailed Description Text (49):
SQL (STRUCTURED QUERY LANGUAGE): A standardized language for defining, querying, maintaining, and protecting the contents of a relational database.

Detailed Description Text (50):
SQL-DATABASE: A relational database made up of tables and views.

Detailed Description Text (55):
SYNCHRONIZATION--WITHIN ONE COMPLETE AUDIT FILE: In an RDB system, the level of synchronization achieved when an audit file is transferred to the remote host immediately following an audit file switch at the source host.

Detailed Description Text (56):
TAKEOVER: In an RDB system, the process that enables the remote database to assure the role of the source database.

Detailed Description Text (57):
TARGET HOST: In an RDB system, the host that contains the remote copy of the database.

Detailed Description Text (60):
There are several remote database (RDB) audit transmission modes and these modes are essentially the key factors which influence how current the second database is aligned with the primary database. The following discussion will discuss the audit block transmission mode, Audit Block Write, (ABW), which is the subject of this invention.

Detailed Description Text (61):
AUDIT BLOCK TRANSMISSION MODE (ABW): The ideal situation of the ABW (Audit Block Write) mode is to transfer audit blocks to the secondary host just as they are generated on the primary host. Under this mode, RDB is able to establish and maintain the greatest degree of synchronization between the primary audit trail and the secondary audit trail thus providing the greatest degree of synchronization of the two databases.

Detailed Description Text (62):
The ABW mode transmits audit data to the secondary host on a block-by-block basis--as it is being written to the audit file on the primary host. The ABW mode makes constant use of the network communications. Network speed and capacity should exceed the audit generation rates to a sufficient degree such that the network does not impede the database throughput. The ABW mode makes the primary host dependent on acknowledgments from the secondary host. The secondary processor speed and

capacity and its disk speed and capacity, must support the audit generation rates so that the secondary host does not impede the response times on the primary host. This mode automatically creates both the original and the duplicate audit files on the secondary host while transferring the audit data only once. Further, this mode operates with one of two possible options for handling problems occurring with audit block transmission.

Detailed Description Text (63):
Utilization of the ABW mode provides certain benefits which include synchronization of the audit trails on the primary and secondary host on a closer basis than is possible with file transfer modes. There is a minimal loss of audit information, which occurs during a disaster or other interruption. Then following a takeover, the restoration of database access is faster than with other modes.

Detailed Description Text (64):
ACKNOWLEDGEMENT RATE: The acknowledgment rate is the rate at which the secondary host sends its acknowledgments to the primary host to indicate receipt of the audit blocks. The acknowledgment rate is set when the user defines the database characteristics for the primary and secondary hosts. A higher acknowledgment rate results with fewer demands on the network, with less risk of communication error and potentially less wait time between audit block transmissions resulting in a faster throughput.

Detailed Description Text (66):
ACKNOWLEDGEMENT AND AUDIT TRAIL SYNCHRONIZATION: The acknowledgment rate affects the audit trail synchronization in the audit block transmission (ABW) mode. In addition, the way in which the acknowledgment rate affects the audit trail synchronization depends on whether the audit files are "sectioned." There are non-sectioned audit files and sectioned audit files, which can be described as follows:

Detailed Description Text (67):
(a) Non-sectioned audit files: here the acknowledgment rate is defined as one acknowledgment message for every n audit blocks and the value n can be set from 1 through 99. The default value is 10. The audit trail synchronization is within 2*n minus 1 audit blocks, unless the secondary database is dropped. Here the * represents a multiplication operation.

Detailed Description Text (68):
(b) Sectioned audit files: the RDB system attempts to acknowledge every n audit blocks where n is the acknowledgment rate. The RDB software rotates the acknowledgment through the sections so that the same section does not always read the acknowledgment.

Detailed Description Text (70):
BUFFERS: In the DMSII XE software, the audit trail BUFFERS option will specify how many internal audit buffers are to be allocated when the database is running. If the BUFFERS option is not specified, then AUTOMATIC is the default value. Under AUTOMATIC, the Accessroutines automatically calculates the number of buffers to be ten times the number of sections declared for the audit trail plus one. For example, if the audit trail has eight sections, then eighty-one buffers are allocated unless otherwise specified.

Detailed Description Text (71):
SECTIONS: The SECTIONS option specifies the number of sectioned files into which the audit trail is to be divided, The default value is 1 (a single audit file, unsectioned). The value can be an integer in the range of 1 thru 63.

Detailed Description Text (72):
Dividing the audit trail into several sectioned files allows the I/O operations to

the audit trail to be spread across several files. Sectioning of the audit trail, along with an improved internal locking and buffering scheme, can help relieve any audit trail bottlenecks impeding overall database throughput. Sectioning allows groups of audit blocks to be transferred on a concurrent parallel operation.

Detailed Description Text (73):
Each audit file is divided into a number of physical audit files designated by the SECTIONS option. The first section of an audit file retains a particular naming convention as follows:

Detailed Description Text (74):
(i) <database name>/AUDIT<n>(primary)

Detailed Description Text (75):
(ii) <database name>/2AUDIT<n>(secondary)

Detailed Description Text (76):
AUDIT TRAIL OPTIONS--EFFICIENCY: increasing the audit trail block size decreases the number of I/O operations performed and thus improves database performance. However, a large audit trail block size also increases the amount of memory to be used for the audit buffers.

Detailed Description Text (77):
UPDATE EOF (END OF FILE): This is an attribute which controls the important trade-off in database performance. Small values for the update EOF option will reduce the number of disk read operations needed to locate the end of the audit trail during recovery. However, more write operations are performed to maintain the end-of-file pointer in block 0 during normal operation of the database.

Detailed Description Text (83):
In general, the present system relates to the situation of providing computer systems which will recall changes to its database in order to allow proper recovery of the database in the event of any failure. Operationally, there is used what is called a transaction, which is a set of related operations that change the content of a database from one particular state to another.

Detailed Description Text (84):
However, before a transaction can currently commit its changes to a database, it is necessary that information about the database rows or records that are affected by the translation be written to what is called an audit trail. An audit trail can be conceived as a history of changes to a database. Such audit trial may consist of a series of files having records which describe changes to the database. Thus, an audit trail record typically consists of a before and an after image of a modified database record.

Detailed Description Text (85):
Using the before images, the database system can undo incomplete modifications which occur when an application program aborts or fails to complete due to a system failure.

Detailed Description Text (86):
Utilizing after images, the database system can recover from media failures by restoring the old or inconsistent copies of database files and redoing the earlier modifications.

Detailed Description Text (88):
This particular method of utilization to the physical storage of audit trail files does involve certain disadvantages. A process that is storing newly generated audit records must then compete for disk access with the archiving of filled audit files. This leads to contention which can limit the rate of audit generation and the

transaction processing speed.

Detailed Description Text (90):
The present invention relates to a method of expediting asynchronously received multiple packets of audit data from a source database host in a resynchronization mode and then asynchronously writing the data at a target host. The audit trail for such a database ordinarily consists of a contiguous order of audit files with no physical partitions. The resynchronization mode of such a target host normally consisted of one physical process to receive multiple contiguous logical audit blocks and to write the blocks in a serial manner in order to duplicate the source audit trail. However, the presently developed transfer operation between source and target hosts is accomplished via sectioned audit files in a concurrently parallel transfer operation as seen in FIG. 9, after initiation by a Tracker sensing unit which indicates an out-of-sync condition.

Detailed Description Text (91):
With the advent of physically partitioning logical audit files containing contiguously ordered audit blocks stored in a round-robin manner to multiple physical files (partitions), multiple asynchronous audit writes are enabled which can then result in faster audit generation at a source database host.

Detailed Description Text (92):
The prior normal physical process in the resynchronization mode required additional processing at the remote host in order to replicate the original audit trail, and this resulted in a process that was slower in duplicating a partitioned audit trail when this was compared to duplicating a non-partitioned audit trail. In this regard, if multiple logical audit blocks could be received asynchronously in a resynchronization mode for each physical partition of a logical audit file and then written asynchronously to corresponding physical files, then the time elapsed in the resynchronization mode was to be substantially reduced.

Detailed Description Text (93):
DATA FLOW UNDER AUDIT BLOCK TRANSMISSION MODE (ABW): With reference to FIG. 3, the normal flow of audit blocks in the ABW mode is illustrated. With the ABW mode, audit block images are transmitted from the database stack 14 through the ACR-PORT I/O Task 22 which is processed from the RDB Support Library 20 on the primary host 10 by way of the ACR_PORT port file 10p, 30p, to the Audit server task 32 on the secondary host, 30.

Detailed Description Text (94):
The Audit server task 32 on the secondary host 30 then writes the audit block images to an audit file 34 on disk. Tracker later reads these audit files from disk and applies these audit block images to the secondary database, 38.

Detailed Description Text (99):
(ii) When an acknowledgment is required, the write waits for a message acknowledgment from the Audit server task 32 on the secondary host, 30.

Detailed Description Text (103):
(b) The Audit server task 32 reads the write operation from the corresponding port file 30p on the secondary host 30.

Detailed Description Text (104):
(c) when required, the Audit server task 32 writes an audit block acknowledgment to the ACR port file on the secondary host and the network 26 transmits the audit block acknowledgment from the secondary host 30 to the primary host 10.

Detailed Description Text (106):
The audit block acknowledgment confirms only that the Audit server task 32 has received the audit block. Waiting for audit block acknowledgment might impose a

delay on the auditing process in the primary host, 10. However, this is the only way to confirm that the audit blocks are present on the secondary host 30.

Detailed Description Text (108):
(i) The Audit server task 32 writes the audit block to the audit file 34 on disk;

Detailed Description Text (110):
PRIMARY DATABASE FOR UPDATE: When the primary database is first opened for an update, the following sets of actions will take place:

Detailed Description Text (111):
(a) The RDB Support Library 20 on the primary host 10 initiates an Audit server task 32 on the secondary host 30 for each section of the audit file. Then steps b, c, d, e are repeated for each section of the audit file. Sectioning was earlier discussed in the General overview portion.

Detailed Description Text (113):
(c) The Accessroutines 32 stops further database activity until the Audit server 32 on the secondary host 30 acknowledges receipt of the first audit block for that section;

Detailed Description Text (114):
(d) The RDB Support Library 20 on the primary host 10 receives the acknowledgment and informs the Accessroutines 12 for the primary database.

Detailed Description Text (117):
FIG. 1 shows how the remote database (RDB) components work together under the ABW mode. There is seen the primary host 10 and the secondary host 30. These two hosts communicate through network connections described later.

Detailed Description Text (118):
When the user opens up the primary database 14, the RDB Support Library 20 is invoked. The RDB Support Library in turn initiates the audit server 32 on the secondary host, 30.

Detailed Description Text (119):
The ACR-PORT I/O Task 22 then takes the audit images from the primary database 14 on the primary host 10 and transfers these images to the Audit server 32 on the secondary host, 30.

Detailed Description Text (120):
Under the ABW audit transmission mode, the audit server 32 then writes the images to the secondary database audit trail 34. The Audit server 32, with the RDBSUPPORT program and the Catchup, FIG. 3, then maintain synchronization of the two audit trails. The Tracker 36 on the secondary host maintains synchronization of the two databases by applying the audit images from the audit trail to the secondary database, 38.

Detailed Description Text (121):
ABW MODE TASKS FOR SECTIONED AUDIT FILES: On the primary host 10, the ACR port I/O task 22 is responsible for sending audit blocks to the secondary host through a dedicated sub-port of the ACR port. On the secondary host 30, the audit server 32 receives audit blocks and writes them to the appropriate audit section. A database schema defines a section as well as all other physical database attributes. The system generates one ACR port I/O task and one audit server task for each audit section. These tasks are always present on both the primary and secondary hosts to provide swift response in the event of a takeover, i.e., switching the job function of secondary host to that of primary host.

Detailed Description Text (122):

TRACKER: The Tracker 36 is an asynchronous remote database (RDB) task program declared and processed from the Data Base Stack 14. The Tracker task appears on either host as:

Detailed Description Text (123):
<database name>/TRACKER.

Detailed Description Text (124):
Tracker is initiated when (i) the database is opened at either the primary or secondary host; (ii) audit images are received at the secondary host; (iii) the RDB-agent detects that a Catchup process 28, 31, of FIG. 3, is necessary; (iv) a RDB utility acknowledgment is performed.

Detailed Description Text (126):
(a) On the secondary host 30, Tracker 36 reads the audit images from the audit trail and applies these images directly to the secondary database 38 through a mechanism similar to a rebuild recovery mechanism. Tracker does not reprocess transactions.

Detailed Description Text (127):
The reading and applying of audit images occurs in two separate phases. During the first phase, known as pre scanning, Tracker reads the audit file looking for a point at which no transactions are in progress. Such a point is known as a quiet point.

Detailed Description Text (128):
During the second phase, Tracker begins to apply all audit images from its current position in the audit trail to the quiet point found during the pre-scanning phase. In other words, Tracker applies audit images from transactions to the secondary database. Tracker does not apply actual transactions.

Detailed Description Text (129):
(b) On the primary host 10, Tracker is always initiated by the first database opener. In most cases, Tracker quickly goes to the end of task (EOT). If a halt/load recovery is needed, however, Tracker waits for the DMRECOVERY task to complete, and then applies any audit afterimages required by the recovery before going to EOT.

Detailed Description Text (130):
(c) Under the ABW audit transmission mode, Tracker initiates the Catchup task (FIG. 3) as soon as it reads to the end of the audit trail at the secondary host and it detects that the primary and secondary audit trails are not synchronized. The Audit Block Serial Numbers (ABSN) are used to determine whether the primary and secondary are out of synchronization. These numbers are logically assigned in a round robin fashion to audit blocks in each section before they are transmitted by the source host. The RDB Support Library on each host maintains a list of ABSN values that are globally accessible by any processes that are involved in the transmission of audit blocks between the two hosts. The tracker on the secondary host communicates with the RDB server on the source host to keep track of ABSN values that have been transmitted. As soon as an out of synchronization condition is detected, the tracker is able to determine the ABSN number of the audit block from which re-transmission has to start, by accessing the last successfully received ABSN value from the RDB Support Library.

Detailed Description Text (131):
TRACKER AND INQUIRY PROGRAMS WORK TOGETHER: Because of database integrity constraints, Tracker must have exclusive use of either database when it is applying audit images. Consequently, when Tracker 36 is applying audit images, inquiry programs are locked out of the database. Conversely, when inquiry programs are accessing the database, Tracker is not able to apply audit images.

Detailed Description Text (132):
Lockout Time is the time that users are unable to read data from a database because Tracker may be in the process of changing the database.

Detailed Description Text (133):
Inquiry programs and Tracker lock out each other from the database only during the time that Tracker applies audit images. The length of the lockout time is dependent on the contents of the audit trail; the time also varies by site of database. Tracker does not lock out inquiry programs while Tracker is pre-scanning the audit trail.

Detailed Description Text (134):
If the Tracker task does not terminate normally, it locks out all inquiry programs when it resumes applying audits to the database until it comes to a point where the database is in a consistent state. At that point, Tracker again allows inquiries while it is pre-scanning audits.

Detailed Description Text (135):
Each time Tracker comes up, the inquiry programs are locked out until Tracker can ensure the integrity of the secondary database.

Detailed Description Text (136):
CATCHUP AND CATCHUP-SERVER: The Catchup 31 and the Catchup Server Tasks 28 operate only when the RDB system is functioning under the ABW audit transmission mode. Their combined functions are called the audit synchronization process. This process is designed to bring the secondary database audit trail back into synchronization with a primary database audit trail when the former (secondary database) is behind the latter (primary database audit trail).

Detailed Description Text (137):
The Catchup and Catchup Server Tasks are part of the Catchup process shown in FIG. 3. This operates in the following way:

Detailed Description Text (139):
If a communication problem prevents the Catchup process from initiating immediately, then the RDB-agent task (discussed below) which is given the name <database name>/RDB/AGENT, attempts communication with the other host at the following types of intervals: (i) on the primary host, one minute following an audit transmission error and every five minutes thereafter; (ii) on the secondary host, every five minutes following an audit transmission error.

Detailed Description Text (141):
(b) The Catchup Server Task 28 reads audit blocks on the primary host and sends these blocks to the Catchup task 21 on the secondary host.

Detailed Description Text (142):
The Catchup Server Task appears on the primary host as:

Detailed Description Text (143):
<database name>/CATCHUP/SERVER/<secondary host name>.

Detailed Description Text (144):
(c) The Catchup task 21 operates on the secondary host and writes to the audit pack the incoming audit blocks sent by the Catchup Server Task 28. The Catchup task 31 also acknowledges receipt of the audit blocks. The Catchup task appears on the secondary host as:

Detailed Description Text (145):
<database name>/CATCHUP.

Detailed Description Text (146):
(d) The Catchup task 31 communicates with the Catchup Server Task 28 to determine when the Catchup process is complete.

Detailed Description Text (152):
CHARACTERISTICS OF PORT port FILE: The PORT port file is used to communicate status information while the database is open or the RDB utility is running. The PORT port file transfers the RDB utility information between the RDB server and the RDB Support Library 20. This port file has the following characteristics:

Detailed Description Text (154):
(ii) This port file only closes when the RDB Support Library 20 for the database goes to the end of task (EOT), or when there is a communication error.

Detailed Description Text (157):
(b) Traffic on this port file is directly proportional to the primary database audit generation;

Detailed Description Text (159):
The Database Stack 14 causes the ACR_PORT port file to be opened during a database open operation. An ACR_PORT port file task appears in the mix as:

Detailed Description Text (160):
<database name>/ACRPORTIO.

Detailed Description Text (165):
SECTIONED AUDIT FILES AND PORT FILES: The RDB system transfers audit images from "sectioned" audit files through sub-ports of each port file (one sub-port for each section). The sub-port (ACR-PORT) used is the section number plus 3. For example, in an audit file with three sections, the name of the task for the third section would be:

Detailed Description Text (166):
<database name>/ACRPORTIO/2

Detailed Description Text (167):
The present discussion will refer to the audit transmission mode (ABW). In operation of the audit block write (ABW) situation, then when an audit block is written to the audit file 18 on the primary host, it is automatically transmitted by way of the network 26, FIG. 3, to the audit file 34 on the secondary host. It is, of course, desirable to get the best possible audit trail synchronization as near as possible to real time. For the best case scenario, this may work out to within one audit block of non-synchronization on a non-sectioned audit file with an acknowledgment rate value of 1.

Detailed Description Text (168):
When the system transfers audit images block-by-block, the data in the remote database operation (RDB) is considered to be backed-up when the audit records for that have been copied from the primary host to the secondary host. However, at this point, the information in the audit trail has not yet been applied to the secondary database. Therefore, it can be seen that the primary and secondary databases are not necessarily synchronized even though their audit trails may be synchronized with respect to the audit files 18 in the primary host, and 34 in the secondary host.

Detailed Description Text (169):
As an example, if the system should happen to run the same inquiry on a newly-updated record on both the primary and secondary hosts simultaneously, there will, of course, be a retrieval of different answers if the updated particular record is

still remaining in the audit trail of the secondary host and has not yet been applied to the secondary database 38. The remote database (RDB) software ensures that the primary database 14 and the secondary database 38 are synchronized by applying the audit images to the secondary database as they are received on the secondary host.

Detailed Description Text (170):
SYNCHRONIZATION LEVELS: The level of database audit trail synchronization that is chosen is tied onto two key factors of database recovery operations: (i) the amount of time required to reestablish the database access following any interruption; and (ii) the amount of data that will be lost as a result of such an interruption.

Detailed Description Text (171):
If there is a secondary database audit trail 34 that is synchronized with its primary database audit trail 18, then when the primary database audit trail 18 becomes available, the system is in a good position to recover the database quickly with a minimal loss of data. It is also a good situation to recover operations quickly and with a minimal loss of data if the RDB is operated at a delayed level of synchronization. The reason for this is because there is a database already set-up to take over the operations of the primary database, so it is then possible to apply outstanding audits as quickly as possible and still be back online for normal operations in a minimal length and predictable length of time.

Detailed Description Text (172):
In operating under a particular level of synchronization, it is necessary to consider the impact of the fact of losing data should the primary database become unavailable. Thus, the more closely synchronized that the audit trails 18 and 34 are, then the smaller amount of data that could be lost should a primary host failure occur.

Detailed Description Text (173):
The network with the databases, the hosts, and the workload and performance involve a tightly integrated system of operations. A heavy workload on any one component can impact a performance of the other components. For example, heavy network traffic can cause a degradation in database performance. Thus in this case, the more closely synchronized the audit trails 18 and 34 are, then, the more sensitive the environment becomes to heavy workloads placed on any one of the components.

Detailed Description Text (174):
The present system involves a method of expediting the transfer of asynchronously received multiple packets of audit data from a source database host 10 in a resynchronization (Catchup) mode and asynchronously writing the data onto a target host 30. The audit trail for such a database normally consisted of a continuous order of audit files (18, 34) with no physical partitions.

Detailed Description Text (176):
In the advent of physically partitioning the logical audit files containing contiguously ordered audit blocks stored in a round-robin manner (FIG. 6) to multiple physical files (partitions), then multiple asynchronous audit writes are enabled which result in a faster audit transfer from the source database host 10 to the secondary host 30.

Detailed Description Text (177):
There is one physical process in the resynchronization mode (Catchup) which requires additional processing at the remote host 30 in order to replicate the original audit trail 18. This results in a process that is slower in duplicating a partitioned audit trail when compared to duplicating a non-partitioned audit trail. In this respect, however, if multiple logical audit blocks could be received asynchronously in a resynchronization (Catchup) mode for each physical partition of a logical audit file and then written asynchronously to corresponding physical

files, then the time elapsed in the resynchronization mode could be considerably reduced.

Detailed Description Text (178):
Performance improvement is achieved by asynchronously receiving multiple packets of audit data from a source database host 10 over a communications network 26 and then asynchronously writing the audit data to multiple physical files 34 at a target host 30. To implement this performance improvement, the so-called logical synchronization process is referred to as the Catchup and consists of multiple physical Catchup processes which are executed automatically by the back-up database system at the remote host 30. Initially, the back-up system recognizes that the resynchronization process (Catchup) is required, and from its shared database library task (RDB Support Library 20, 35) will then initiate one physical Catchup task for each physical audit file partition.

Detailed Description Text (179):
The method by which the secondary host recognizes that the resynchronization process is required consists of a TRACKER process 36 running at the secondary host that reads the Audit blocks which it uses to update the Database. When an End Of File is reached reading the audit file, the tracker checks with the Primary host 10 for a loss of synchronization. If a loss of synchronization is detected, the tracker 36 at the secondary host 30 causes an event called START_CATCHUP to its RDB Support Library 35. The WAITANDRESET process running continuously within the RDB Support Library monitors for the START_CATCHUP event to be caused by the tracker process. This is shown in FIG. 2 where the Primary RDB Support Library 20 sends Global ABSN values to the Secondary RDB Support Library 35. The START_CATCHUP event is recognized by the WAITANDRESET process, which then builds, for each section of the audit file, the parameters of the point from which resynchronization is needed. These parameters are further discussed in detail in FIG. 6. The WAITANDRESET process, acting on the causing of the START_CATCHUP event passes these parameters to execute the CATCHUP processes, initiated for each section of the audit file. FIG. 2 indicates the internal and exported processes running from the RDB Support Library during the resynchronization process.

Detailed Description Text (180):
The first physical Catchup task is one that is responsible for opening the port files of all Catchup and initiating the server task at the source host Database Stack 14 by calling through the database library (RDB Support Library 20). The Catchup server task at the source host 10 reads logical audit blocks in the order in which they were written and stores multiple blocks in parallel buffers, one per partition, for transfer to the remote host 30. These buffers are located in the Catchup Server Task 28, FIG. 3.

Detailed Description Text (181):
The parallel buffers are only used in FIG. 3 by the Catchup Server Task 31. FIG. 1 depicts normal audit transfer (i.e., database activity when no Catchup is needed).

Detailed Description Text (182):
Each time a Catchup task receives a group of contiguous physical audit blocks, it writes them to the corresponding physical file (audit file 34). Each time the Accessroutines server 12 fills a buffer in Catchup Server 28, then a write of the buffer is initiated to the secondary host 30 and the server 12 continues reading the logical audit blocks and storing them in the buffers intended for the Port Writes (ports 10p, 30p) (FIG. 1) until another buffer is full and another Write is initiated.

Detailed Description Text (183):
This process repeats itself until the server 12 determines that the source and remote audit trails (18 and 34) are synchronized.

Detailed Description Text (186):
An overall view of the presently-described synchronization process is shown in FIG. 4. Referring to step (A), the Tracker (36, FIG. 3) at the remote host 30 will read the audit blocks and work to apply to the secondary database 38.

Detailed Description Text (188):
At step (C), the Catchup task 31 opens the port files 30c and initiates a Catchup server task 28 in the primary host 10.

Detailed Description Text (189):
At step (D), the Catchup Server Task 28 in the primary host 10 reads the audits and writes them to a series of buffers.

Detailed Description Text (191):
At step (F), the Catchup task 31 at the remote host 30 will receive the audit data from the buffers and Write the data to multiple physical audit files 34 in the remote or secondary host 30.

Detailed Description Text (193):
Referring to FIG. 5, at step (A1), the Tracker task program 36 will read the audit files 34 and then at step (A2) the Tracker will write the audit data out to the secondary database 38.

Detailed Description Text (196):
At step (A4), if the answer NO, that is to say no Catchup is needed, then the sequence proceeds to step (A4N) where the tracker waits for the next audit blocks and reads the audit blocks for transfer to the secondary base 38 from the audit file 34.

Detailed Description Text (197):
In summary, the Catchup operation will substantially expedite the synchronization process. A Catchup operation is needed following a network failure or transmission delay because once the network connection is reestablished, the audit trails are out of sync between the source and remote host. A process is needed that is able to transfer all necessary audit data faster than the rate at which it is being generated. In the past, non-sectioned Catchup did this by having Catchup Server 28 read multiple logical audit blocks into one large buffer before sending it through a port file to a remote host Catchup task. In the present usage for sectioned audit, catchup Server 28 behaves similarly in reading the audit trail, but speeds up the process by storing the multiple logical blocks in large parallel buffers, 1 per audit section and sending each buffer through its own port file to its matching Catchup task.

Detailed Description Text (199):
There is one Catchup task for each section of audit. Each task is initiated simultaneously. Each task performed reeds from its own port connection which corresponds to one audit section and then writes the data to the matching physical audit section at the secondary. The first Catchup task (there will always be at least one) has the responsibility of initiating the Catchup Server Task at the primary sides.

Detailed Description Text (205):
At step (C3), the first Catchup task 28 initiates a server task at the source host 10 via the RDB Support Library 20.

Detailed Description Text (207):
At step (C5), the Catchup task operations 31 will asynchronously write multiple audit files received from the source host over to the multiple physical files 34 at the remote host 30.

Detailed Description Text (208):
FIG. 8 is a flow chart illustrating the sequence of step operations involved for the Catchup Server Task unit 28 of FIG. 3.

Detailed Description Text (209):
At step (CS1), the Catchup Server Task unit 28 at the source host 10 will open the ports 10c, 30c.

Detailed Description Text (210):
At step (CS2), the Catchup Server Task 28 at the primary source host 10 then reads the logical audit blocks in the order in which they were written. This is done from the audit file 18 (FIG. 3).

Detailed Description Text (211):
At step (CS3), the Catchup Server Task unit 28 then stores multiple blocks in parallel buffers, with one buffer for each partition for enabling transfer to the remote host 30. The use of the parallel buffers enabling the transfer of a buffer for each partition thus provides a parallel set of transfer operations which speeds up the transfer of multiple audit blocks to the secondary host 30.

Detailed Description Text (212):
At step (CS4), there is an asynchronous writing of the parallel buffers to the remote host 30, which is initiated by the Catchup Server Task 28.

Detailed Description Text (213):
At step (CS5), the Catchup Server Task unit 28 repeats the reading of audit blocks and continues storing them in multiple buffers, then writing them to the remote host 30 asynchronously until the Catchup server 28 determines that the source audit trail in audit file 18 is synchronized with the receipt audit file 34 in the secondary host 30 after which it then stops (END) the process of transferring audit blocks.

Detailed Description Text (214):
It should be seen that a "partition" is illustrated in FIG. 9, sections 18a, 18b and 18c. "Section" is a logical splitting of audit files into segregated blocks. Further, the audit blocks which are received from the source are read asynchronously by the Catchup task unit 31 in a fashion which bypasses any serial reading of the logical blocks but rather in a parallel fashion from the multiple Catchup tasks which are utilized by tracker 36 for storage in the secondary database 38 in their logical order.

Detailed Description Text (215):
Thus, by the use of parallel transfer modes for transferring the logically partitioned audit files, there is a more efficient transfer of audit blocks at a higher speed in order to expedite the synchronization process.

Detailed Description Text (216):
Referring to FIG. 9, there is shown a schematic diagram which illustrates the parallel and concurrent transfer of sectioned audit files from the primary host to the secondary host or remote host.

Detailed Description Text (217):
The remote Database Support Library 35 communicates with the Catchup Server Task program 28 which, for illustrated purposes, is seen to have several sectioned audit files designated 18a, 18b,and 18c.

Detailed Description Text (218):
Then by use of the Catchup port file 10c, 30c, FIG. 3 (CU_PORT port file), these sectioned audit files (18a, 18b, 18c,) are transferred over to the secondary host audit file and indicated as audit files 34a, 34b, and 34c.

Detailed Description Text (219):
Thus, in cooperation with the first Remote Database Support Library 20 and the secondary Remote Database Library 35, the Catchup Server 28 can now transfer the sectioned audit files as multiple sectioned groups to be attended to by a series of Catchup process operations (31a, 31b, 31c) which work together in parallel to asynchronously transfer the sectioned audit files to the Tracker 36 in the secondary host for placement onto the secondary database 38.

Detailed Description Text (220):
While earlier systems used only a single serial transfer mode for audit block conveyance to the secondary host, the present system now enables multiple sectioned audit files to be simultaneously and concurrently transferred to the secondary host for placement on the secondary database 38.

Detailed Description Text (221):
As a result, the secondary (backup) database 38 is maintained in useful synchronism with the primary (source) database 14. That is to say, that now the secondary database accurately reflects the data in the primary database at any given time period.

Detailed Description Text (222):
Described herein has been a method and system for increasing the transfer rate of audit trails of a Primary source host for placement to a remote secondary host to enable a secondary database to serve as backup to a primary database. A Tracker program senses the level of synchronization between the audit files of the primary host and secondary host to determine if there is sufficient lag (out of synchronization) between primary and secondary audit trails. If this is the case, a Catchup program is invoked which divides the audit trail into sections which are expeditiously transferred in a parallel asynchronous basis from primary to secondary host for placement on a secondary backup database.

CLAIMS:

1. A Catchup process providing a method for increasing the transfer rate of audit blocks from a primary host to a secondary host when a lack of synchronism in audit data is indicated, said method comprising the steps of:

(a) initiating, via a library support unit, the start of a Catchup mode for expediting transfer of packets of audit blocks;

(b) opening up, via a Catchup Task unit, specialized port files for connection from said primary host to said secondary host;

(c) signaling, by said Catchup Task unit, to said library support unit to start up a Catchup server unit in said primary host;

(d) asynchronously receiving, by said secondary host, of multiple packets of said audit blocks concurrently along parallel paths;

(e) placing said received multiple packets into physical files at said secondary host.

4. A method for asynchronous parallel transfer of sectioned audit files received at a secondary remote host (30) from a primary source host (10) when a condition of non-synchronization is indicated, said method comprising the steps of:

(a) initiating, by a Database Support Library means, of a sequential set of Catchup operations;

(b) opening special Catchup port files to enable sectioned audit files from said primary source host to deposit said sectional audit files at said secondary remote host;

(c) utilizing said sequential set of Catchup operations to concurrently and asynchronously write said multiple sets of sectioned audit files to said remote host.

5. A system for expediting transfer of audit blocks of sectioned audit files read into parallel buffers at a primary host over to sectioned audit files at a secondary host when a condition of non-synchronization between said primary host and secondary host is indicated, said system comprising:

(a) Catchup server task means for reading audit blocks from a sectioned audit file and placing them in multiple buffers, each buffer of which will hold a group of blocks from a corresponding audit file section of said primary source audit file;

(b) library support means for conveying each buffer section of said primary audit trail into multiple sections of secondary remote audit files;

(c) multiple sets of Catchup programs operating in concurrent sequences onto said remote sets of sectioned audit files for expediting their transfer onto a secondary database.

6. In a network for transferring audit files from a primary host to a secondary host when a lack of synchronism occurs between said primary host and said secondary host, a Catchup system for expediting the transfer of sectioned audit files from said primary host to said secondary host comprising:

(a) library support means for initiating a first Catchup task to expedite the delivery of a first audit file section onto said remote database;

(b) means to open up the port files of all Catchup tasks to enable a steady flow of sectioned audit files from said primary host onto said secondary host;

(c) Catchup server means for initiating a sequence of port write operations, each operation of which will transfer a sectioned audit file of said primary host onto said database of said secondary host in a parallel concurrent set of operations.

L16: Entry 5 of 12 File: USPT Sep 3, 2002

DOCUMENT-IDENTIFIER: US 6446090 B1
TITLE: Tracker sensing method for regulating synchronization of audit files between primary and secondary hosts

Abstract Text (1):
A method for sensing and regulating the level of synchronization between a file of audit blocks generated in a primary host for transfer to a secondary host. The audit files in a primary host are divided into audit blocks which are each identified with an audit block serial number. The individual audit blocks are transmitted through a network from a primary host to a secondary host wherein a tracker mechanism monitors the difference between the number of received audit blocks and the number of generated audit blocks residing at the primary host waiting to be transferred. When the disparity between the audit blocks received and the generated audit blocks waiting to be transmitted reaches a certain critical level, the tracker mechanism will initiate an advance speed up program to expedite the transfer in order to bring the primary and secondary audit files into greater synchronism. Further, the acknowledgment level can be set to indicate the number of audit blocks received in the secondary host for information back to the primary host and the acknowledgment level can be utilized in order to preset a critical value which will indicate that the speed up transfer operation should be initiated.

Parent Case Text (2):
This application is related to co-pending applications U.S. Ser. No. 09/374,351 entitled: "Enhanced System and Method For Management of System Database Utilities," and co-pending application U.S. Ser. No. 09/415,273 entitled: "System And Method For Asynchronously Receiving Multiple Packets Of Audit Data From A Source Database Host In A Resynchronization Mode And Asynchronously Writing The Data To A Target Host," which are incorporated herein by reference.

Brief Summary Text (2):
This disclosure relates to network communications where logical audit blocks are created at a source host and transferred to a remote host where the audit trail is used to create and maintain a continuously synchronized remote database backup.

Brief Summary Text (4):
A database such as the Unisys Data Management System II, Extended, is a centralized collection of data placed into one or more files. Multiple application programs can access this data concurrently. Consequently, redundant files are not required for each individual application. Application programs running in batch, time sharing, and reload job entry environments can all access the database concurrently. A database of the present configuration consists of the following major components: (a) Data sets; (b) Sets; (c) Subsets; (d) Data items; (e) Global data.

Brief Summary Text (5):
A data set, a set, or a subset, that is not an item of another set is termed disjoint. Structures need not be disjoint, that is to say a hierarchy can exist between the various data sets, sets, and subsets. A data set, a set, or a subset, that is an item in another data set, is said to be embedded. When a database contains embedded structures, a hierarchical file structure results.

Brief Summary Text (7):
A set is a structure that allows access to all records of a data set in some logical sequence. The set contains one entry for each record in the data set. Each set entry is an index that locates a data set record. If key items are specified for the set, records in the data set are accessed based upon these keys. Otherwise the records are accessed sequentially. Multiple sets can be declared for a single data set, thereby enabling the data in a data set to be accessed in several different sequences. A subset is similar to a set. Unlike a set, a subset need only refer to selected records in the data set. A data item is a field in a database record used to contain an individual piece of information.

Brief Summary Text (8):
Data items that are not a part of any data set are then called global data items. Global data items generally consist of information such as control totals, hash totals, and populations, which apply to the entire database. All global data items are stored in a single record.

Brief Summary Text (9):
The audit trail is a record of changes made to the database. The audit trail is used to recover automatically the database following a hardware or software failure. The audit trail specification clause describes the physical attributes of the audit trail.

Brief Summary Text (10):
The audit trail, as mentioned, consists of a record of changes to the database. It is only created for audited databases and is used in the various forms of database recovery. An audit trail specification describes the attributes of the audit trail. The specification is optional. If no specification appears, attributes are assigned by default. All audited databases must include a "restart" data set definition. There is a specialized syntax for specifying the audit trail attributes. These involve area size, area length, block size, buffers, checksum, and sections in addition to whether disk or tape is involved and types of tape being used.

Brief Summary Text (13):
A Remote Database Backup or RDB is a database recovery system which can be a key component of a disaster recovery plan since it minimizes the amount of time needed to recover from a loss of database access. The RDB system also minimizes the loss of productivity, minimizes the loss of revenue and minimizes the loss of business, which could occur because of interruptions in the ability to access one's database. The RDB works in conjunction with the Data Management System II (DMSII) databases plus Structured Query Language Database (SQLDB), the Semantic Information Manager (SIM) database, and the Logic and Information Network Compiler II (LINCII) databases.

Brief Summary Text (14):
The components of the RDB system consist of a database and also a copy of the database. One database is update capable and the other database can be used only for inquiry purposes. The update-capable database is called the primary database. The host on which this database resides is called the primary host. The "current on-line" remote database copy, which is called the secondary database, is "inquiry-capable" only. The host on which this database resides is called the secondary host. The configuration of the primary and the secondary databases on their separate hosts is called the RDB System. A single host can participate in multiple RDB systems.

Brief Summary Text (15):
The RDB or remote database backup system enables users to maintain a current on-line inquiry-only copy of a database on an enterprise server, which is separate from the enterprise server on which the update-capable database resides. The host

locations can be at the same site or at two geographically distant sites. The remote database backup keeps the database copy up-to-date by applying the audit images from the audited database to the database copy. There is a choice of four audit transmission modes which enables one to choose the means of audit transfer between hosts.

Brief Summary Text (16):
In the RDB system, the term "primary" and the term "secondary" will indicate the intended function of each copy of the database and the host on which it resides.

Brief Summary Text (17):
The primary database has the function for database inquiry and update, while the secondary database has the functionality useful for database inquiry only.

Brief Summary Text (18):
The secondary database cannot be updated by any application programs and the secondary database is modified only by the application of audit images of transactions performed on the primary database.

Brief Summary Text (19):
Since one complete RDB system is made of one database, and includes the secondary database which resides on another host, that is to say the primary database on one host plus one copy of that database.

Brief Summary Text (20):
A host is the system on which a primary or a secondary database resides. A host can function as a primary host in one RDB system and then also concurrently function as a secondary host for another RDB system. Additionally, one host can function as a secondary host (or a primary host) for multiple RDB systems.

Brief Summary Text (21):
When a RDB system is first initialized for a database, then by default, the primary host is the host upon which the database resides. The other host which is defined for that database is designated as a secondary host and it remains a secondary host until a takeover is performed or until the RDB capability is disabled. Both the primary and secondary hosts must have sufficient resources to support the RDB system and its application environment.

Brief Summary Text (22):
As an illustration, it can be seen how the primary database on a system, which is called Host One and the secondary database is applied on a system called Host Two can work together in response to or in anticipation of an interruption on the primary host. In this example, the application normally runs against the primary database in Host One with the RDB transferring audit images to the secondary database. Under normal operation, which is when the audit images are transferred from the primary database to the secondary database without loss of data during transmission due to network or system failure, the example described above works well. However, in the condition that a network or system failure results in the loss of data during transmission from the primary database to the secondary then the secondary database is said to be out of synchronization with the primary database. Hence there is need of a mechanism by which the lost data can be re-transmitted so that the secondary database can be re-synchronized with the primary one.

Brief Summary Text (23):
The object of the instant invention is to provide a sensing and regulation mechanism between a primary host and a secondary host wherein sectioned audit files established as audit blocks are organized for transfer from a primary host through a network communications bus over to a secondary host with the object of eventually using the received audit block files to update a remote database to keep it in

synchronization with a database in the primary host.

Brief Summary Text (24):
In order to accomplish this, there is provided a tracker program and mechanism which is made sensitive to the number of audit blocks in the primary waiting to be transferred to the secondary compared with the number of audit blocks actually received in the secondary which will be used to update the secondary database. Due to transmission delays or broken network communication lines, there can develop a very undesirable out of synchronization situation between the audit block data in the primary and the audit block data in the secondary host. Thus the present sensing and regulation mechanism is devoted to sensing this difference gap and regulating it in order to expeditiously provide for a greater synchronization of audit block data between the primary host and the secondary host.

Brief Summary Text (25):
AUDIT TRAIL SYNCHRONIZATION: It is of some importance to decide on what is called audit level synchronization that is desired for the remote database backup system. This involves the question of "how closely must the backup database match its source database? Or to express it in another fashion, how closely synchronized should the secondary database audit trail be a replicate of the primary database audit trail?"

Brief Summary Text (26):
MODES OF AUDIT TRANSMISSION: The remote database backup (RDB) system provides four specific audit transmission modes that enable the user to regulate whether the transmission of the audit images is to be automatic or manual; whether the transmission of audit images is to be done as individual audit blocks or entirely whole audit files; whether the transmission of audit images can be interrupted, that is to say, suspended or not; and what is to be the degree of audit trail synchronization between the primary host and the secondary host. The focus of the present invention involves the use of one mode designated as the ABW or Audit Block write mode.

Brief Summary Text (27):
AUDIT BLOCK WRITE (ABW): The secondary audit trail is to be constantly and automatically kept synchronized with the primary database audit trail on a block-by-block basis. The ABW mode enables this type of close synchronization level to occur by (i) handling interruptions to audit transmissions through one of two error handling options; or (ii) initiating a Catch-up process for the audit block transfer whenever the usual synchronization level is disrupted. This invention is devoted to the Catch-up process.

Brief Summary Text (30):
In a system wherein audit files are transferred from a primary source host through a network connection over to a secondary target backup host, it is essential to sense and regulate any disparity between the audit data in the primary host and the secondary host so that the secondary host does not lag too far behind duplicating the audit data that resides in the primary host.

Drawing Description Text (5):
FIG. 4 is a flow chart showing the steps involved in the audit trail transport process between a primary and secondary database with the use of the Catchup process for expedition of synchronization;

Drawing Description Text (9):
FIG. 8 is a flow chart illustrating the Catch-up Server Task for reading audit blocks and writing them asynchronously to the remote host until the audit trails are synchronized;

Detailed Description Text (1):

GLOSSARY LIST ACKNOWLEDGMENT RATE: The rate at which the secondary host sends an acknowledgment to the primary host to indicate receipt of audit blocks. ACR: Abbreviation for Accessroutines, the software component of the DMSII product that is primarily responsible for the accessing (creating, modifying and deleting) of data in a DMSII database and auditing all changes to the database. AUDIT: An examination of systems, programming and data center procedures in order to determine the efficiency of a computer system. AUDIT DATA: For DMSII databases, data that records every change to a predefined database. AUDIT FILE: For DMSII databases, a file produced by the Accessroutines that contains various control information, including before and after images of records resulting from changes to the database. AUDIT FILE VS. AUDIT BLOCK: For DMSII databases, the audit file represents one or more physical files that contain audit blocks that are stored sequentially. AUDIT FILE SWITCH: For DMSII databases, the logical time when one audit file is complete and a new one is started. AUDIT IMAGES: For DMSII databases, structured package of data representing change to the database that are stored sequentially into the audit trail. AUDIT SOFTWARE: These are specialized programs to perform a variety of auditing functions, such as sampling databases or possibly generating confirmation letters to customers. It can be used to highlight certain exceptions to categories of data and alert the user to possible errors. Audit software may often include a non-procedural language that lets the auditor-user describe the computer and the data environment without need for detailed programming. AUDIT TRAIL: This is a record of transactions in an information system that provides verification of the activity of the system. The simplest audit trail is a transaction itself. For example, if an employee's salary is increased, the changed transaction will include the date, the amount of the raise, and the name of the authorizing manager. It is possible to create a more elaborate audit trail when the system is being verified for accuracy. For example, samples of processing results can be recorded at various stages. Item counts and hash totals can be used to verify that all input has been processed through the system. For DMSII databases, the sequence of audit files that are created and span the life of the database. CATCHUP: In an RDB system, the process that brings the remote audit trail back into synchronization with the source audit trail following a suspension of normal audit transfer. CATCHUP TASK: In an RDB system, a physical process that runs at a remote host, reads audit data from a port file connected to a source database, and writes the data to a physical audit file. DMSII XE: This denotes a Unisys Corporation Data Management System-Extended. FASTER AUDIT GENERATION: For DMSII databases, a rate of audit generation that can be achieved by using sectioned audit and multiple processors. FILE: A collection of bytes which is stored as an individual entity. For example, all data on disk is stored as a file with an assigned file name that is unique within the directory it resides in. To the computer, file is only nothing more than a series of bytes. The structure of a file is known to the software that manipulates it. For example, database files are made up of a series of records. Word processing files (also called documents) contain a continuous flow of text. FILE ATTRIBUTE: A file access classification that allows a file to be retrieved or erased. Typical attributes are read/write, read only, archive, and hide or hidden. FILE FORMAT: This is the structure of a file. There are hundreds of proprietary formats for a database, for word processing, and for graphics files. FILE MAINTENANCE: (i) This is the periodic updating of master files. For example, this might include adding/deleting employee names and customer names, or making address changes or changing product prices. This does not refer to daily transaction processing and batch processing, such as order processing and billing and so on. (ii) The periodic reorganization of a disk drive. Data that is continuously updated becomes physically fragmented over the disk space and requires regrouping. An optimization program can be run daily or weekly in order to rewrite all the files on a contiguous basis. FILE MANAGER: (i) This is software that manages a data file and is not to be confused with a database manager. The file managers provide the ability to create, enter, change, query, and produce reports on one single file at a time. There is no relational capability and it does not involve a programming language. (ii) Often used for software used to manage files on a disk. It provides functions to delete, copy, remove, rename, and view files as

well as to create and manage directories. FILE NAME: This is a name assigned by the user or the programmer that is used to identify a particular file. FILE SERVER: This is a high speed computer in the local area network (LAN) that stores the programs and the data files shared by users of the network. Sometimes it's called a network server and it acts like a remote disk drive. LINCII DATABASE: A database generated by the LINC system software; may be a DMKSII database. LOGIC & INFORMATION NETWORK COMPILER (LINC): A software development tool that may be used to generate a DMSII database and any number of applications to access the database. LOGICAL AUDIT BLOCK: For DMSII databases, a structured package containing potentially many Audit Records (in the extreme situation, it is also possible that a single Audit Block could only contain a partial audit Record). LOGICAL AUDIT FILE: For DMSII databases, the sequential storage of Audit Blocks that contain Audit Records. One Logical Audit File may contain 1 or more Physical Audit Files (Sections or Partitions). The sequence of Audit Blocks is spread, round robin fashion, among the Audit Sections. LOGICAL RESYNCHRONIZATION PROCESS (CATCHUP): In an RDB system, the mode of resynchronizing the primary and secondary audit trails following a network failure during normal audit transfer. MASTER CONTROL PROGRAM: This is the operating system which runs and regulates Accessroutines. NON-PARTITIONED AUDIT FILE: In a DMSII system, an audit file that has one section or partition. Equally, an audit file that contains one physical file. NON-SECTIONED AUDIT FILES: Same as NON-PARTITIONED AUDIT FILES. NORMAL AUDIT TRANSFER: In an RDB system, the uninterrupted transfer of audit data from a source database host to a remote host while the source database is being updated. ORIGINAL AUDIT TRAIL: In an RDB system, the audit trial of the source database. PACKET (OF AUDIT DATA): For DMSII databases, a collection of one or more audit blocks. PARALLEL BUFFERS: Any number of storage areas each of the same size. PARTITIONED AUDIT FILE: For DMSII databases, a logical audit file that is partitioned into a predefined number of physical files. PERIODIC SYNCHRONIZATION: In an RDB system, audit synchronization that takes place only when complete audit files become available for transfer to a remote host (i.e., following an audit file switch). PHYSICAL AUDIT FILE: A physical file containing Audit Blocks. May be 1 of many sections of a Logical Audit File. PORT FILE NETWORK COMMUNICATION: In an RDB system, the method of messaging and data transfer between a source database system and a remote backup system. REMOTE DATABASE BACKUP: A disaster recovery capability for DMKSII-based databases that enables the replication of an audit (primary) database on a secondary host. The replicated (secondary) database is kept up-to-date with the primary database through the application of audits from the primary database. when the primary database becomes unavailable, the secondary database can take over the role of the primary database. REMOTE HOST: In an RDB system, the host that contains the duplicate copy of the source database. Also known as the Secondary Host. RDB SYSTEM: (Remote Database Backup): This is a Unisys Corporation system for backup of a database and is referenced by a Unisys Publication Item 8600-2052-304 dated December, 1998, entitled "Remote Database Backup--Planning and Operations Guide." RDBSUPPORT LIBRARY: In an RDB system, the library that is accessed by the shared task, database utilities, and additional applications responsible for configuring an RDB system. The library is also a running process responsible for initiating local and remote tasks through port file communication. RDB UTILITY: The menu-driven user interface for defining, installing, and maintaining an RDB system. RESYNCHRONIZATION MODE: Under the ABW audit file transmission mode of an RDB database, the process of bringing the audit trail of the secondary database back into the closest possible synchronization with the audit trial of the primary database. Also see Catchup. SECTIONAL AUDIT FILES: Same as PARTITIONED AUDIT FILES. SEMANTIC INFORMATION MANAGER (SIM): A database management system that simplifies the task of modeling your application environment based on the semantic data model. SERVER TASK: In an RDE system, a task that is connected to a remote host for messaging and data transfer. SHARED DATABASE TASK: For DMSII databases, the running process accessed by all database applications to read and write data to the database and audit trail. SIM-DATABASE: A DMSII database defined by SIM. SOURCE DATABASE HOST: In an RDB system, the host that contains the primary copy of the database. SQL (STRUCTURED QUERY LANGUAGE): A standardized language for defining,

querying, maintaining, and protecting the contents of a relational database. SQL-DATABASE: A relational database made up of tables and views. SYNCHRONIZED AUDIT TRAILS: In an RDB system, audit trails at a source and remote host that are exact duplicates. SYNCHRONIZATION LEVEL: In an RDB system, the level at which the remote audit trail is kept current as a replicate of the source audit. SYNCHRONIZATION (NEAR REAL TINE): In an RDB system, the level of synchronization achieved when each audit block is transferred to the remote host immediately after it is written at the source host. SYNCHRONIZATION (PRESENT CONTEXT): In an RDB system, the process of updating a remote audit trail to replicate the source audit trail. SYNCHRONIZATION--WITHIN ONE COMPLETE AUDIT FILE: In an RDB system, the level of synchronization achieved when an audit file is transferred to the remote host immediately following an audit file switch at the source host. TAKEOVER: In an RDB system, the process that enables the remote database to assure the role of the source database. TARGET HOST: In an RDB system, the host that contains the remote copy of the database. TRACKER: This is a specialized program which operates to observe the number of audit blocks received by a secondary host and contrasts this with a number of audit blocks residing in the primary host which are yet to be transferred to the secondary host. When a certain critical value of the contrast between the two sets of audit block files is reached, then the tracker program will institute a speed up process to expedite the transfer of audit blocks from the primary host to the secondary host until a more desirable level of parity and duplication occurs between the audit files in the primary host and the secondary host.

Detailed Description Text (3):
There are several remote database (RDB) audit transmission modes and these modes are essentially the key factors which influence how current the second database is aligned with the primary database. The following discussion will discuss the audit block transmission mode, Audit Block Write, (ABW), which is the subject of this invention.

Detailed Description Text (4):
AUDIT BLOCK TRANSMISSION MODE (ABW): The ideal situation of the ABW (Audit Block Write) mode is to transfer audit blocks to the secondary host just as they are generated on the primary host. Under this mode, RDB is able to establish and maintain the greatest degree of synchronization between the primary audit trail and the secondary audit trail thus providing the greatest degree of synchronization of the two databases.

Detailed Description Text (5):
The ABW mode transmits audit data to the secondary host on a block-by-block basis as it is being written to the audit file on the primary host. The ABW mode makes constant use of the network communications. Network speed and capacity should exceed the audit generation rates to a sufficient degree such that the network does not impede the database throughput. The ABW mode makes the primary host dependent on acknowledgments from the secondary host. The secondary processor speed and capacity and its disk speed and capacity, must support the audit generation rates so that the secondary host does not impede the response times on the primary host. This mode automatically creates both the original and the duplicate audit files on the secondary host while transferring the audit data only once. Further, this mode operates with one of two possible options for handling problems occurring with audit block transmission.

Detailed Description Text (6):
Utilization of the ABW mode provides certain benefits which include synchronization of the audit trails on the primary and secondary host on a closer basis than is possible with file transfer modes. There is a minimal loss of audit information, which occurs during a disaster or other interruption. Then following a takeover, the restoration of database access is faster than with other modes.

Detailed Description Text (7):
ACKNOWLEDGMENT RATE: The acknowledgment rate is the rate at which the secondary host sends its acknowledgments to the primary host to indicate receipt of the audit blocks. The acknowledgment rate is set when the user defines the database characteristics for the primary and secondary hosts. A higher acknowledgment rate results with fewer demands on the network, with less risk of communication error and potentially less wait time between audit block transmissions resulting in a faster throughput.

Detailed Description Text (9):
ACKNOWLEDGMENT AUDIT TRAIL SYNCHRONIZATION: The acknowledgment rate affects the audit trail synchronization i the audit block transmission (ABW) mode. In addition, the way in which the acknowledgment rate affects the audit trail synchronization depends on whether the audit files are "sectioned." There are non-sectioned audit files and sectioned audit files, which can be described follows: (a) Non-sectioned audit files: here the acknowledgment rate is defined as one acknowledgment message for every n audit blocks and the value n can be set from 1 through 99. The default value is 10. The audit trail synchronization is within 2*n minus 1 audit blocks, unless the secondary database is dropped. Here the * represents a multiplication operation. (b) Sectioned audit files: the RDB system attempts to acknowledge every n audit blocks where n is the acknowledgment rate. The RDB software rotates the acknowledgment through the sections so that the same section does not always read the acknowledgment. The audit trail synchronization will generally be within 2*n minus 1 blocks but actually could be a higher value, up to the number of audit sections. For example, if the number of audit sections is 3, the audit trail synchronization would be within (2*n minus 1) plus 3 blocks.

Detailed Description Text (10):
BUFFERS: In the DMKSII XE software, the audit trail BUFFERS option will specify how many internal audit buffers are to be allocated when the database is running. If the BUFFERS option is not specified, then AUTOMATIC is the default value. Under AUTOMATIC, the Accessroutines automatically calculates the number of buffers to be ten times the number of sections declared for the audit trail plus one. For example, if the audit trail has eight sections, then eighty-one buffers are allocated unless otherwise specified.

Detailed Description Text (11):
SECTIONS: The SECTIONS option specifies the number of sectioned files into which the audit trail is to be divided. The default value is 1 (a single audit file, unsectioned). The value can be an integer in the range of 1 thru 63. Dividing the audit trail into several sectioned files allows the I/O operations to the audit trail to be spread across several files. Sectioning of the audit trail, along with an improved internal locking and buffering scheme, can help relieve any audit trail bottlenecks impeding overall database throughput. Sectioning allows groups of audit blocks to be transferred on a concurrent parallel operation.

Detailed Description Text (12):
Each audit file is divided into a number of physical audit files designated by the SECTIONS option. The first section of an audit file retains a particular naming convention as follows: (i) <database name>/AUDIT<n>(primary) (ii) <database name>/2AUDIT<n>(secondary)

Detailed Description Text (13):
AUDIT TRAIL OPTIONS--EFFICIENCY: Increasing the audit trail block size decreases the number of I/O operations performed and thus improves database performance. However, a large audit trail block size also increases the amount of memory to be used for the audit buffers.

Detailed Description Text (14):
UPDATE EOF (END OF FILE): This is an attribute which controls the important trade-

off in database performance. Small values for the update EOF option will reduce the number of disk read operations needed to locate the end of the audit trail during recovery. However, more write operations are performed to maintain the end-of-file pointer in block 0 during normal operation of the database.

Detailed Description Text (18):
In general, the present system relates to the situation of providing computer systems which will recall changes to its database in order to allow proper recovery of the database in the event of any failure. Operationally, there is used what is called a transaction, which is a set of related operations that change the content of a database from one particular state to another.

Detailed Description Text (19):
However, before a transaction can currently commit its changes to a database, it is necessary that information about the database rows or records that are affected by the transaction be written to what is called an audit trail. An audit trail can be conceived as a history of changes to a database. Such audit trial may consist of a series of files having records which describe changes to the database. Thus, an audit trail record typically consists of a before and an after image of a modified database record.

Detailed Description Text (20):
Using the before images, the database system can undo incomplete modifications which occur when an application program aborts or fails to complete due to a system failure.

Detailed Description Text (21):
Utilizing after images, the database system can recover from media failures by restoring the old or inconsistent copies of database files and redoing the earlier modifications.

Detailed Description Text (23):
This particular method of utilization to the physical storage of audit trail files does involve certain disadvantages. A process that is storing newly generated audit records must then compete for disk access with the archiving of filled audit files. This leads to contention which can limit the rate of audit generation and the transaction processing speed.

Detailed Description Text (25):
The present invention relates to a tracker mechanism which involves a method of surveying the status of audit block files which have been received at a secondary host and comparing them with a series of audit block files which are waiting in the primary host to be transferred to the secondary host due to delays and sometimes inoperability of the network communication lines between the primary and secondary host. An extreme lack of synchronization can occur whereby the audit blocks in the secondary host no longer duplicate the audit blocks in the primary host and thus lead to a condition which is denoted as non-synchronization of the audit files. The presently-described tracker method can use the audit block write serial numbers which have been placed to identify each audit block in the primary host which is waiting to be transferred. These audit block serial numbers can be compared with the highest audit block serial number of the audit block received at the secondary host and thus there can be a calculation of the disparity as to how much lack of duplication is occurring as between the secondary host in duplicating the primary host audit files. This amount of disparity can be regulated or set to a certain level so that when a certain level of disparity occurs, then the tracker program will initiate another program which will speed up the transfer process but only if there is a sufficient lack of disparity as between the audit block files of the secondary host in relationship to the primary host.

Detailed Description Text (26):

With the advent of physically partitioning logical audit files containing contiguously ordered audit blocks stored in a round-robin manner to multiple physical files (partitions), multiple asynchronous audit writes are enabled which can then result in faster audit generation at a source database host.

Detailed Description Text (27):
The prior normal physical process in the resynchronization mode required additional processing at the remote host in order to replicate the original audit trail, and this resulted in a process that was slower in duplicating a partitioned audit trail when this was compared to duplicating a non-partitioned audit trail. In this regard, if multiple logical audit blocks could be received asynchronously in a resynchronization mode for each physical partition of a logical audit file and then written asynchronously to corresponding physical files, then the time elapsed in the resynchronization mode was to be substantially reduced.

Detailed Description Text (28):
DATA FLOW UNDER AUDIT BLOCK TRANSMISSION MODE (ABW): With reference to FIG. 1, the normal flow of audit blocks in the ABW mode is illustrated. With the ABW mode, audit block images are transmitted from the database stack 14 through the ACR-PORT I/O Task 22 which is processed from the RDB Support Library 20 on the primary host 10 by way of the ACR_PORT port file 10p, 30p, to the Audit server task 32 on the secondary host, 30.

Detailed Description Text (29):
The Audit server task 32 on the secondary host 30 then writes the audit block images to an audit file 34 on disk. Tracker 36 later reads these audit files from disk and applies these audit block images to the secondary database, 38.

Detailed Description Text (30):
INITIATION OF DATA TRANSMISSION: The creation of an audit block image initiates the data transmission from the primary host 10 to the secondary host 30. This data transmission is part of a DMSII audit block write (ABW) operation that includes several items: (a) The logical (direct I/O) write to the audit disk file 18 on the primary host 10. This logical write waits for the completion of the physical write to disk. (b) The logical (port file I/O) write to the ACR_PORT port file (10p, 30p) that leads to the secondary host 30. This logical write waits for an event that indicates the completion of the port file I/O write, as developed below: (i) The write always waits for a write result from the MCP indicating that the port file I/O write has occurred. (ii) When an acknowledgment is required, the write waits for a message acknowledgment from the Audit server task 32 on the secondary host, 30.

Detailed Description Text (32):
OPERATIONAL STEPS FOR THE PORT FILE I/O WRITE OPERATION: The steps that complete the port file write operation (FIG. 1) are indicated below as follows: (a) The network 24 transmits the write operation from the primary host 10 to the secondary host 30. When an audit block acknowledgment is required, the primary RDB Support Library 20 indicates the requirement by setting a field in the port file I/O message. (b) The Audit server task 32 reads the write operation from the corresponding port file 30p on the secondary host 30. (c) when required, the Audit server task 32 writes an audit block acknowledgment to the ACR port file on the secondary host and the network 24 transmits the audit block acknowledgment from the secondary host 30 to the primary host 10. (d) When an acknowledgment is requested, the primary host writes and transmits n-1 more audit blocks, where (n is the acknowledgment rate) before it reads the acknowledgment from the secondary host. The audit block acknowledgment confirms only that the Audit server task 32 has received the audit block. Waiting for audit block acknowledgment might impose a delay on the auditing process in the primary host, 10. However, this is the only way to confirm that the audit blocks are present on the secondary host 30.

Detailed Description Text (33):
SECONDARY HOST HANDLING OF THE AUDIT BLOCK: After receiving the audit block and sending an audit block acknowledgment, when required, the following actions will occur on the secondary host 30: (i) The Audit server task 32 writes the audit block to the audit file 34 on disk; (ii) Tracker 36 reads the audit block for transfer to secondary database 38.

Detailed Description Text (34):
PRIMARY DATABASE FOR UPDATE: When the primary database is first opened for an update, the following sets of actions will take place: (a) The RDB Support Library 20 on the primary host 10 initiates an Audit server task 32 on the secondary host 30 for each section of the audit file. Then steps b, c, d, e are repeated for each section of the audit file. Sectioning was earlier discussed in the General Overview portion. (b) The Accessroutines 12 writes the first audit block on the primary host; (c) The Accessroutines 32 stops further database activity until the Audit server 32 on the secondary host 30 acknowledges receipt of the first audit block for that section; (d) The RDB Support Library 20 on the primary host 10 receives the acknowledgment and informs the Accessroutines 12 for the primary database. (e) The Accessroutines 12 completes the audit block write process and allows the processing to continue.

Detailed Description Text (36):
FIG. 1 shows how the remote database (RDB) components work together under the ABW mode. There is seen the primary host 10 and the secondary host 30. These two hosts communicate through network connections described later.

Detailed Description Text (37):
When the user opens up the primary database 14, the RDB Support Library 20 is invoked. The RDB Support Library in turn initiates the audit server 32 on the secondary host, 30.

Detailed Description Text (38):
The ACR-PORT I/O Task 22 then takes the audit images from the primary database 14 on the primary host 10 and transfers these images to the Audit server 32 on the secondary host, 30.

Detailed Description Text (39):
Under the ABW audit transmission mode, the audit server 32 then writes the images to the secondary database audit trail 34. The Audit server 32, with the RDBSUPPORT program and the Catch-up, FIG. 3, then maintain synchronization of the two audit trails. The Tracker 36 on the secondary host maintains synchronization of the two databases by applying the audit images from the audit trail to the secondary database, 38.

Detailed Description Text (40):
ABW MODE TASKS FOR SECTIONED AUDIT FILES: On the primary host 10, the ACR port I/O task 22 is responsible for sending audit blocks to the secondary host through a dedicated sub-port of the ACR port. On the secondary host 30, the audit server 32 receives audit blocks and writes them to the appropriate audit section. A database schema defines a section as well as all other physical database attributes. The system generates one ACR port I/O task and one audit server task for each audit section. These tasks are always present on both the primary and secondary hosts to provide swift response in the event of a takeover, i.e., switching the job function of secondary host to that of primary host.

Detailed Description Text (41):
TRACKER: The Tracker 36 is an asynchronous remote database (RDB) task program declared and processed from the Data Base Stack 14. The Tracker task appears on either host as: <database name>/TRACKER. Tracker is initiated when (i) the database is opened at either the primary or secondary host; (ii) audit images are received

at the secondary host; (iii) the RDB-agent detects that a Catch-up process 26, 31, of FIG. 3, is necessary; (iv) a RDB utility acknowledgment is performed.

Detailed Description Text (42):
TRACKER OPERATIONS: Tracker 36 performs a certain number of operations as follows: (a) On the secondary host 30, Tracker 36 reads the audit images from the audit trail and applies these images directly to the secondary database 38 through a mechanism similar to a rebuild recovery mechanism. Tracker does not reprocess transactions. The reading and applying of audit images occurs in two separate phases. During the first phase, known as pre scanning, Tracker reads the audit file looking for a point at which no transactions are in progress. Such a point is known as a quiet point. During the second phase, Tracker begins to apply all audit images from its current position in the audit trail to the quiet point found during the pre-scanning phase. In other words, Tracker applies audit images from transactions to the secondary database. Tracker does not apply actual transactions. (b) On the primary host 10, Tracker is always initiated by the first database opener. In most cases, Tracker quickly goes to the end of task (EOT). If a halt/load recovery is needed, however, Tracker waits for the DMRECOVERY task to complete, and then applies any audit after-images required by the recovery before going to EOT. (c) Under the ABW audit transmission mode, Tracker initiates the Catch-up task (FIG. 3) as soon as it reads to the end of the audit trail at the secondary host and it detects that the primary and secondary audit trails are not synchronized. The Audit Block Serial Numbers (ABSN) are used to determine whether the primary and secondary are out of synchronization. These numbers are logically assigned in a round robin fashion to audit blocks in each section before they are transmitted by the source host. The RDB Support Library on each host maintains a list of ABSN values that are globally accessible by any processes that are involved in the transmission of audit blocks between the two hosts. The tracker on the secondary host communicates with the RDB server on the source host to keep track of ABSN values that have been transmitted. As soon as an out of synchronization condition is detected, the tracker is able to determine the ABSN number of the audit block from which re-transmission has to start, by accessing the last successfully received ABSN value from the RDB Support Library.

Detailed Description Text (43):
TRACKER AND INQUIRY PROGRAMS WORK TOGETHER: Because of database integrity constraints, Tracker must have exclusive use of either database when it is applying audit images. Consequently, when Tracker 36 is applying audit images, inquiry programs are locked out of the database. Conversely, when inquiry programs are accessing the database, Tracker is not able to apply audit images.

Detailed Description Text (44):
Lockout Time is the time that users are unable to read data from a database because Tracker may be in the process of changing the database.

Detailed Description Text (45):
Inquiry programs and Tracker lock out each other from the database only during the time that Tracker applies audit images. The length of the lockout time is dependent on the contents of the audit trail; the time also varies by site of database. Tracker does not lock out inquiry programs while Tracker is pre-scanning the audit trail.

Detailed Description Text (46):
If the Tracker task does not terminate normally, it locks out all inquiry programs when it resumes applying audits to the database until it comes to a point where the database is in a consistent state. At that point, Tracker again allows inquiries while it is pre-scanning audits.

Detailed Description Text (47):
Each time Tracker comes up, the inquiry programs are locked out until Tracker can

ensure the integrity of the secondary database.

Detailed Description Text (48):
CATCHUP AND CATCHUP-SERVER: The Catch-up 31 and the Catch-up Server Tasks 28 operate only when the RDB system is functioning under the ABW audit transmission mode. Their combined functions are called the audit synchronization process. This process is designed to bring the secondary database audit trail back into synchronization with a primary database audit trail when the former (secondary database) is behind the latter (primary database audit trail).

Detailed Description Text (49):
The Catch-up and Catch-up Server Tasks are part of the Catch-up process shown in FIG. 3. This operates in the following way: (a) Whenever Tracker 36 on the secondary host 30 reaches the end of the audit trail, Tracker determines whether the audit trails are still synchronized. If they are not synchronized, then Catch-up is then initiated at the secondary host. If a communication problem prevents the Catch-up process from initiating immediately, then the RDB-agent task (discussed below) which is given the name <database name>/RDB/AGENT, attempts communication with the other host at the following types of intervals: (i) on the primary host, one minute following an audit transmission error and every five minutes thereafter; (ii) on the secondary host, every five minutes following an audit transmission error. The RDB-agent task is an asynchronous task processed from the RDB Support Library 20. This task stays in the mix as long as the RDB software is executing. (b) The Catch-up Server Task 28 reads audit blocks on the primary host and sends these blocks to the Catch-up task 31 on the secondary host. The Catch-up Server Task appears on the primary host as:

Detailed Description Text (50):
<database name>/CATCHUP/SERVER/<secondary host name>. (c) The Catch-up task 31 operates on the secondary host and writes to the audit pack the incoming audit blocks sent by the Catch-up Server Task 28. The Catch-up task 31 also acknowledges receipt of the audit blocks. The Catch-up task appears on the secondary host as: <database name>/CATCHUP. (d) The Catch-up task 31 communicates with the Catch-up Server Task 28 to determine when the Catch-up process is complete. (e) If Catch-up terminates abnormally or unsuccessfully, it then restarts automatically after the synchronization restart interval has elapsed.

Detailed Description Text (52):
CHARACTERISTICS OF PORT port FILE: The PORT port file is used to communicate status information while the database is open or the RDB utility is running. The PORT port file transfers the RDB utility information between the RDB server and the RDB Support Library 20. This port file has the following characteristics: (i) The traffic on this port file is normally intermittent; (ii) This port file only closes when the RDB Support Library 20 for the database goes to the end of task (EOT), or when there is a communication error.

Detailed Description Text (53):
The ACR_PORT port file (10p, 30p) is used only during normal audit transfer operations when the ABW audit transmission mode is set. This file operates with the following characteristics: (a) Messages consist of audit blocks that, as they are filled, are sent from the primary host to the secondary host; (b) Traffic on this port file is directly proportional to the primary database audit generation; (c) During the Catch-up task, this port file can be open, but audit block transfers only occur through the CU_PORT port file, (10c, 30c).

Detailed Description Text (54):
The Database Stack 14 causes the ACR_PORT port file to be opened during a database open operation. An ACR_PORT port file task appears in the mix as: <database name>/ACRPORTIO.

Detailed Description Text (56):
SECTIONED AUDIT FILES AND PORT FILES: The RDB system transfers audit images from "sectioned" audit files through sub-ports of each port file (one sub-port for each section). The sub-port (ACR-PORT) used is the section number plus 3. For example, in an audit file with three sections, the name of the task for the third section would be: <database name>/ACRPORTIO/2

Detailed Description Text (57):
The present discussion will refer to the audit transmission mode (ABW). In operation of the audit block write (ABW) situation, then when an audit block is written to the audit file 18 on the primary host, it is automatically transmitted by way of the network 26, FIG. 3, to the audit file 34 on the secondary host. It is, of course, desirable to get the best possible audit trail synchronization as near as possible to real time. For the best case scenario, this may work out to within one audit block of non-synchronization on a non-sectioned audit file with an acknowledgment rate value of 1.

Detailed Description Text (58):
When the system transfers audit images block-by-block, the data in the remote database operation (RDB) is considered to be backed-up when the audit records for that have been copied from the primary host to the secondary host. However, at this point, the information in the audit trail has not yet been applied to the secondary database. Therefore, it can be seen that the primary and secondary databases are not necessarily synchronized even though their audit trails may be synchronized with respect to the audit files 18 in the primary host, and 34 in the secondary host.

Detailed Description Text (59):
As an example, if the system should happen to run the same inquiry on a newly-updated record on both the primary and secondary hosts simultaneously, there will, of course, be a retrieval of different answers if the updated particular record is still remaining in the audit trail of the secondary host and has not yet been applied to the secondary database 38. The remote database (RDB) software ensures that the primary database 14 and the secondary database 38 are synchronized by applying the audit images to the secondary database as they are received on the secondary host.

Detailed Description Text (60):
SYNCHRONIZATION LEVELS: The level of database audit trail synchronization that is chosen is tied onto two key factors of database recovery operations: (i) the amount of time required to reestablish the database access following any interruption; and (ii) the amount of data that will be lost as a result of such an interruption.

Detailed Description Text (61):
If there is a secondary database audit trail 34 that is synchronized with its primary database audit trail 18, then when the primary database audit trail 18 becomes available, the system is in a good position to recover the database quickly with a minimal loss of data. It is also a good situation to recover operations quickly and with a minimal loss of data if the RDB is operated at a delayed level of synchronization. The reason for this is because there is a database already set-up to take over the operations of the primary database, so it is then possible to apply outstanding audits as quickly as possible and still be back online for normal operations in a minimal length and predictable length of time.

Detailed Description Text (62):
In operating under a particular level of synchronization, it is necessary to consider the impact of the fact of losing data should the primary database become unavailable. Thus, the more closely synchronized that the audit trails 18 and 34 are, then the smaller amount of data that could be lost should a primary host failure occur.

Detailed Description Text (63):
The network with the databases, the hosts, and the workload and performance involve a tightly integrated system of operations. A heavy workload on any one component can impact a performance of the other components. For example, heavy network traffic can cause a degradation in database performance.

Detailed Description Text (64):
The presently described Tracker method for monitoring and regulating the synchronization condition between the primary and secondary hosts operates within the framework of operations for asynchronously receiving multiple packets of audit data from a source database host 10 in a re-synchronization (Catch-up) mode and asynchronously writing the data onto a target host 30. The audit trail for such a database normally consisted of a continuous order of audit files (18, 34) with no physical partitions.

Detailed Description Text (66):
In the advent of physically partitioning the logical audit files containing contiguously ordered audit blocks stored in a round-robin manner (FIG. 6) to multiple physical files (partitions), then multiple asynchronous audit writes are enabled which result in a faster audit transfer from the source database host 10 to the secondary host 30.

Detailed Description Text (67):
There is one physical process in the resynchronization mode (Catchup) which requires additional processing at the remote host 30 in order to replicate the original audit trail 18. This results in a process that is slower in duplicating a partitioned audit trail when compared to duplicating a non-partitioned audit trail. In this respect, however, if multiple logical audit blocks could be received asynchronously in a resynchronization (Catch-up) mode for each physical partition of a logical audit file and then written asynchronously to corresponding physical files, then the time elapsed in the resynchronization mode could be considerably reduced.

Detailed Description Text (68):
Performance improvement is achieved by asynchronously receiving multiple packets of audit data from a source database host 10 over a communications network 26 and then asynchronously writing the audit data to multiple physical files 34 at a target host 30. To implement this performance improvement, the so-called logical synchronization process is referred to as the Catch-up and consists of multiple physical Catch-up processes which are executed automatically by the back-up database system at the remote host 30. Initially, the back-up system recognizes that the resynchronization process (Catch-up) is required, and from its shared database library task (RDB Support Library 20, 35) will then initiate one physical Catch-up task for each physical audit file partition.

Detailed Description Text (69):
The method by which the secondary host recognizes that the resynchronization process is required consists of a TRACKER process 36 running at the secondary host that reads the Audit blocks which it uses to update the Database. When an End Of File is reached reading the audit file, the tracker checks with the Primary host 10 for a loss of synchronization. If a loss of synchronization is detected, the tracker 36 at the secondary host 30 causes an event called START_CATCHUP to its RDB Support Library 35. The WAITANDRESET process running continuously within the RDB Support Library monitors for the START_CATCHUP event to be caused by the tracker process. This is shown in FIG. 2 where the Primary RDB Support Library 20 sends Global ABSN values to the Secondary RDB Support Library 35. The START_CATCHUP event is recognized by the WAITANDRESET process, which then builds, for each section of the audit file, the parameters of the point from which resynchronization is needed. These parameters are further discussed in detail in FIG. 6. The WAITANDRESET

process, acting on the causing of the START_CATCHUP event passes these parameters to execute the CATCHUP processes, initiated for each section of the audit file. FIG. 2 indicates the internal and exported processes running from the RDB Support Library during the resynchronization process.

Detailed Description Text (70):
The first physical Catch-up task is one that is responsible for opening the port files of all Catch-up and initiating the server task at the source host Database Stack 14 by calling through the database library (RDB Support Library 20). The catchup server task at the source host 10 reads logical audit blocks in the order in which they were written and stores multiple blocks in parallel buffers, one per partition, for transfer to the remote host 30. These buffers are located in the Catch-up Server Task 28, FIG. 3.

Detailed Description Text (71):
The parallel buffers are only used in FIG. 3 by the Catch-up Server Task 31. FIG. 1 depicts normal audit transfer (i.e., database activity when no Catch-up is needed).

Detailed Description Text (72):
Each time a Catch-up task receives a group of contiguous physical audit blocks, it writes them to the corresponding physical file (audit file 34). Each time the Accessroutines server 12 fills a buffer in Catch-up Server 28, then a Write of the buffer is initiated to the secondary host 30 and the server 12 continues reading the logical audit blocks and storing them in the buffers intended for the Port Writes (ports 10p, 30p) (FIG. 1) until another buffer is full and another Write is initiated.

Detailed Description Text (73):
This process repeats itself until the server 12 determines that the source and remote audit trails (18 and 34) are synchronized.

Detailed Description Text (76):
An overall view of the synchronization process is shown in FIG. 4. Referring to step (A), the Tracker (36, FIG. 3) at the remote host 30 will read the audit blocks and work to apply to the secondary database 38.

Detailed Description Text (78):
At step (C), the Catch-up task 31 opens the port files 30c and initiates a Catchup server task 28 in the primary host 10.

Detailed Description Text (79):
At step (D), the Catch-up Server Task 28 in the primary host 10 reads the audits and writes them to a series of buffers.

Detailed Description Text (81):
At step (F), the Catch-up task 31 at the remote host 30 will receive the audit data from the buffers and Write the data to multiple physical audit files 34 in the remote or secondary host 30.

Detailed Description Text (83):
Referring to FIG. 5, at step (A1), the Tracker task program 36 will read the audit files 34 and then at step (A2) the Tracker will write the audit data out to the secondary database 38.

Detailed Description Text (86):
At step (A4), if the answer NO, that is to say no Catch-up is needed, then the sequence proceeds to step (A4N) where the tracker waits for the next audit blocks and reads the audit blocks for transfer to the secondary base 38 from the audit file 34.

Detailed Description Text (87):
In summary, the Catch-up operation will substantially expedite the synchronization process. A Catch-up operation is needed following a network failure or transmission delay because once the network connection is reestablished, the audit trails are out of sync between the source and remote host. A process is needed that is able to transfer all necessary audit data faster than the rate at which it is being generated. In the past, non-sectioned Catch-up did this by having Catch-up Server 28 read multiple logical audit blocks into one large buffer before sending it through a port file to a remote host Catch-up task. In the present usage for sectioned audit, Catch-up Server 28 behaves similarly in reading the audit trail, but speeds up the process by storing the multiple logical blocks in large parallel buffers, 1 per audit section and sending each buffer through its own port file to its matching Catch-up task. These port writes occur asynchronously which speeds up the operation. The process is further sped up by each Catch-up task writing to its corresponding physical audit section asynchronously.

Detailed Description Text (88):
There is one Catch-up task for each section of audit. Each task is initiated simultaneously. Each task performed reads from its own port connection which corresponds to one audit section and then writes the data to the matching physical audit section at the secondary. The first Catch-up task (there will always be at least one) has the responsibility of initiating the Catch-up Server Task at the primary side.

Detailed Description Text (94):
At step (C3), the first Catch-up task 28 initiates a server task at the source host 10 via the RDB Support Library 20.

Detailed Description Text (96):
At step (C5), the Catch-up task operations 31 will asynchronously write multiple audit files received from the source host over to the multiple physical files 34 at the remote host 30.

Detailed Description Text (97):
FIG. 8 is a flow chart illustrating the sequence of step operations involved for the Catch-up Server Task unit 28 of FIG. 3.

Detailed Description Text (98):
At step (CS1), the Catch-up Server Task unit 28 at the source host 10 will open the ports 10c, 30c.

Detailed Description Text (99):
At step (CS2), the Catch-up Server Task 28 at the primary source host 10 then reads the logical audit blocks in the order in which they were written. This is done from the audit file 18 (FIG. 3).

Detailed Description Text (100):
At step (CS3), the Catch-up Server Task unit 28 then stores multiple blocks in parallel buffers, with one buffer for each partition for enabling transfer to the remote host 30. The use of the parallel buffers enabling the transfer of a buffer for each partition thus provides a parallel set of transfer operations which speeds up the transfer of multiple audit blocks to the secondary host 30.

Detailed Description Text (101):
At step (CS4), there is an asynchronous writing of the parallel buffers to the remote host 30, which is initiated by the Catch-up Server Task 28.

Detailed Description Text (102):
At step (CS5), the Catch-up Server Task unit 28 repeats the reading of audit blocks

L16: Entry 4 of 12 File: USPT Mar 25, 2003

DOCUMENT-IDENTIFIER: US 6539402 B1
TITLE: Using periodic spaces of block ID to improve additional recovery

Brief Summary Text (2):
This invention relates to the field of audit trail storage and recovery and has particular application to systems of auditing databases.

Brief Summary Text (4):
Large or rapidly accessed database performance in real time has become a business tool of necessity in communications, electronic commerce, and as support for processes in many other forms of commerce. Thus, the ability to recover quickly from a system or partial system failure has become a weak link in the chain of support for computing and communications systems which run the data bases to support commerce and communications. The importance of quick recovery is underscored by the fact that many systems have been designed to allow operations during recovery by an audit system. Such a system is described in U.S. Pat. No. 5,734,817, issued to Roffe et al., and its disclosure incorporated herein in its entirety by this reference.

Brief Summary Text (5):
Currently, many database servers have tape storage audit trails, and the tape systems are typically running very quickly, say, filling a tape in 12 minutes or less, to accommodate large amounts of data needed for recovery. The records are typically stored in "audit blocks" of fixed or variable length, depending on the system, and several thousand of them can be stored per second. The tape and other audit systems will also typically have system status saves stored on regular intervals selected by the audit program or audit program manager. These records may be stored in the form of audit blocks, (and we call such blocks "P-Saves" for "periodic saves" of system data for the remainder of this document).

Brief Summary Text (6):
Thus, it is required to search through the audit blocks in order to find from where to begin the reconstruction process so that a database can be restored to its state prior to the crash. This restoration will cause any records which may have been opened, or opened and partially operated upon, or opened, partially operated upon left not closed, to be set to an appropriate state or any exceptions issued where necessary. (A transaction process that is completed is sometimes called a completed "step". A step is a term that will be used frequently herein each of these operations would be viewed as such a "step". The importance of this term will become apparent within.) It is easy to understand that such reconstruction and restoration are critical functions in financial transactional databases. Thus, for a bank or other commercial operation to be unable to accomplish such restoration work very quickly is anathema to their business success since all operations of such a compromised database should be put on hold until the recovery is complete. To do otherwise would be to risk the credibility of the data integrity in the whole system, and the business (or other operations) which depends on such records being accurate.

Brief Summary Text (7):
The tape storage systems which contain most modem audit data are typically

optimized to run forwards while making recordings at a rapid continuous rate, and consequently, actually operating them to recover from a failure instead requires them to run in non optimized modes, introducing delays in recovery time which can be hours long. Part of the delay is introduced in reading each audit data block, determining what is in it, then backing up (reversing) the tape to the next previous block, reading it, determining what is in it, and so on, until all open items or incomplete transactions are discovered. Only then can a reasonable system proceed to read the entire tape forward to once again read the audit blocks and reconstruct the activity occurring at the time of the failure so that the failure can be corrected or appropriately accommodated. Such wait times before beginning recovery can be extremely significant, risking the real time commercial or communications activities for which the databases are used. Also, positioning near the end of the tape may be time consuming due to the size of the latest tape storage systems. However, positioning near the end of the tape for many of these latest tape storage systems is required to start a search for all activity in progress at the time of the system failure.

Brief Summary Text (10):
In the context of using mass storage for audit trail information, U.S. Pat. No. 5,561,795 (Sarkar), incorporated herein by this reference, describes a system of keeping a time for the beginning of a transaction (that is, one affecting each of several cached pages of a database) and storing it updated every time the cache audit trail is being written to the non-volatile (mass) storage. Sarkar requires a transaction control table from which the oldest uncommitted transaction can be found. In a system with thousands of on-going transactions, therefore each one would require entries in this table in order for it to be useful in establishing an audit trail in accord with Sarkar's invention.

Brief Summary Text (13):
Accordingly there is great commercial need for a method or system to speed up examination of audit record blocks preparatory to restoration of a database to a fully operative condition, and at the same time does not require much if any storage area in main memory or processing overhead to implement.

Drawing Description Text (8):
FIG. 6 is a block diagram of an example transaction process using a database.

Drawing Description Text (11):
A system and method are described providing for expediting restoration and/or recovery in situations where a transactional system or database may require resort to audit trail records. It employs an asymmetrically nested location indicator system that can be employed by an executive recovery program to get to the earliest "step" that has not been committed to prior to the crash (or other event requiring a reconstruction or recovery). It can be thought of as a system employing a series of opportunely located and conveniently generated pointers to the start of any open action items (uncommitted steps and uncompleted steps), thus enabling a recovery executive program to begin recovery operations expeditiously, without requiring maintenance in main memory or in any cache a record of the location and/or time of each step that is active and completed.

Detailed Description Text (3):
Accordingly, refer now to FIG. 1 which illustrates the logical organization of audit blocks of memory in a typical mass storage system which can be used with this invention. In a typical system a limited number of audit files in mass storage, such as files 11, 12, 13, and 14, are available for the storage of the audit trail in that mass storage. At the beginning of each file an identifier is stored. (Here the identifier is illustrated as 1, 2, n(32), and (eoL). It should be noted that in systems produced by the assignee of this patent, a file is also called an "F-cycle"). The identifier should identify the name of the file and/or its address on the mass storage medium. In an exemplary system the number of audit files available

in the rotating queue is on the order of about 32 although hundreds of files or more may be allocated for this activity if required under certain circumstances. Thus, in this illustration the file numbered 13 is 30 audit files away from the file 12. Through various available schemes this logical distance can be accommodated if desired in other mass storage as audit trail files on mass storage are usually transferred to a more permanent media, for example, tape, after a specific amount of time to free up the mass storage for more audit trail files. The end of each audit file in the series will have a pointer to the beginning of the next file in the series. Typically, these files can vary in size between less than 100 words to hundreds of thousands of words. If the end of the series of blocks is reached and the programmer or controller of the audit trail feature is set up to do it, the programmer or controller may allocate other space in other mass storage devices or tape reader systems into which these blocks of data may be stored longer term, therefore extending the length of the audit trail indefinitely if desired. One method and system for writing audit records to an area of non-volatile memory is described in U.S. patent application Ser. No. 09/001,136 (RA-5075), in the application entitled, XPC BACKUP FOR IN-PROGRESS AUDIT, by Cooper, Hill, Konrad and Nowatzki, and assigned to the assignee hereof, and which is hereby incorporated by this reference in its entirety. Another system for storing audit information which includes both a master and a slave outboard memory device is described in U.S. Pat. No. 5,940,826, and failure features of such a system are described in U.S. Pat. No. 5,949,970; both of which are also incorporated herein in their respective entireties, by this reference. Thus, it should be recognized that there are diverse systems for storing audit trail information.

Detailed Description Text (4):
In FIG. 2 a series 20 of tape cartridges 21, 22, and 23, are shown having identification numbers 135, 605, and n, respectively. The previous tape's number in a series of audit trail tapes is recorded at the beginning of each subsequent tape in the series, and the next tape in the series is identified at the end of the preceding tape cartridge. Thus, the entire series of data recorded across the series of tapes can be located in sequence similarly to the way data is recorded in audit trail files as described above. Here, a tape numbered 132 (P=#132 may be recorded) was previous in the recording series to the tape numbered 135. Thus, 135 is recorded at the start of tape 132. For the next tape 605, the number 135 is recorded since 135 was directly preceding tape 605. And the tape n follows the tape 605 as can be seen in the P numbers and N numbers identified at the bottom of the Figure. Likewise at the end of each tape, the next tape numbers shall be written, this having "605" at the end of tape 135; "n" at the end of 605, and so on. Thus, the large data files in mass storage in the series or the tapes themselves need to be organized such that they each contain a header block and a trailer block with a pointer to the previous end and next tape or file, respectively. In FIG. 3 a large magnetic tape 30 (which could also represent a file of memory storage on a magnetic disk drive) is seen to contain a header block 33 and a trailer block 38 identifying the previous file on mass storage and the next file on mass storage, or the previous tape and the next tape, respectively. Thus, in the tape example, the P (previous tape) number is 132. The time that the tape was started will also be recorded in the header block as information 25. Time data may be used for other purposes as well in perfecting a recovery of data or finding a spot in an audit trail, and may be used to enhance recovery operations. The system will store data in Audit Blocks, like blocks 34 and 35 illustrated here. There will likely be a very large number of blocks like 34 and 36 in any given file like tape 132 illustrated here or one of the files like 11 of FIG. 1. There will be a relatively small number of P-SAVE records like 35 and 37. (In Audit block 35, the P-SAVE record takes up the whole audit block since in the preferred embodiment, we will begin a next audit block when a P-SAVE is made. A more typical audit block with a P-SAVE record would be like that in block 37, where the block size is only somewhat truncated with the occurrence of a P-SAVE. Clearly, one can organize these blocks as is convenient for the particular system, even designating unique block types for P-SAVES if desired without going outside the scope of the ideas presented here).

The P-SAVE records will only be generated at particular times selected by the executive program maintaining the audit trail. Depending on the number of transactions or size of the modifications being made or both, the executive (or a programmer/user) may select different amounts of time between storage of P-SAVE blocks. The first block of data, Data Block 34 in the data record 31 on tape 30, contains the type of information kept in an information record, here in the information record identified with numeral 26. Here, a file called A was updated by making changes to records identified by numbers 1-7 and 9 and the data from these records is also stored. (These data can be before-look or after-look, it matters not for the purposes of this discussion. The data from these records may be the data that was stored previous to the transaction (or step) or after the transaction (or step) depending on the way the programmer has set up the system. For some detail on what before look, that is, previous to the transaction, and after look, that is, after the transaction, records contain and how such a system works, refer to U.S. Pat. No. 5,682,527, Cooper, et al., incorporated herein in its entirety by this reference. In most systems expected to use this invention only before-look or after-look records will be stored in the audit trail. The Sarkar reference appears to require or expect storage of both before and after-look records. Use of either a before-look or after-look only audit system like in the Cooper reference or of a system that stores both would be supported by our inventions.) Thus, a complete record of the transactions occurring in a database is being kept.

Detailed Description Text (5):
In the next block of data 35, here a P-SAVE block, the audit trail system has (in a preferred system) stored a time stamp for when this P-SAVE was made. It will contain an indication of which steps are active, including what commits are in progress, and what rollbacks may be in progress at this time stamp P-SAVE point. (In preferred embodiment systems we do not keep rollbacks, but this may be desirable depending on the methods applied to restoration of a database chosen). All this information may be wanted to recover from a failure. (The name or other identifier of the step will be recorded only since the actual information was already stored in the audit blocks.) The P-SAVE does not store data on the time and location of the earliest currently active step. It will also contain locations and time data too, preferably for the last previous P-SAVE. FIG. 3 will be referred back to later to describe more details of how the invention operates.

Detailed Description Text (7):
Referring now to FIG. 4, in which such a system 40 is illustrated, the database 41 is here assumed to be in a single location although distributed databases may also enjoy the fruits of this invention. A manager program 46 operates on the data in the database system in accord with requests from remote 42 and local 43 user information systems. The series of communications links and systems through which a remote user may employ the database is here illustrated simply as a cloud 44. Any operation that a user might employ to change data in the database would be considered a step. Because steps occur between the manager program and the users, steps are illustrated as area 45. The manager program 46 will log every activity it takes in relation to the data in the database 41 by recording it through an audit information record manager program 47. This information record manager program 47 will simply organize the data so that it appears in a manner consistent with that illustrated in FIG. 3 when it is recorded there. (There may be cache records made in an intermediate time frame for other types of recovery if desired, this is not relative to this invention.)

Detailed Description Text (8):
If there is a failure in the operation of the system which requires a recovery, an executive recovery program 39 must be available to manipulate either or all of the database information, the program manager, and the audit information manager in order to effectuate a recovery. The audit information manager 47 will control its own records 19 in preferred embodiments.

Detailed Description Text (9):
A process for getting to the point where the executive recovery program 39 could force a repair of the data in the database 41 so that none of the open transactions (open steps) would be lost, is illustrated in FIG. 5A. The process 80 assumes continuous periodic saves of transaction information and P-SAVE data by the audit information record manager 47 blocks of data, and the P-SAVES make a periodic saving of the state of any open, also known as active, Steps. In general terms, a Step is started when a user or a process requests that something be done to some data within the database. A Step ends when that something that was started to be done or requested to done is completed and the record in the database being operated on is closed. Thus the process step 81 of continuously auditing database transaction Steps continues until some failure mode 82 is detected. Assuming that the system wants the recovery to proceed, the first step is to step backward through the audit trail to find the last P-SAVE 83. The data in the P-SAVE will, as previously described, contain the identity of each Step that is currently active and any commits in progress. ("A Commit in Progress" would be where a request has been made to apply the database updates. In other words, the Step has requested the system to apply the updates). It could also contain the identity of any rollbacks in progress if the system is tracking these (a rollback is a process of a user operating on a record which has been temporarily committed to but wishes to change the temporary commitment before making a full commitment or restoring the values to their current status without making the change). A rollback is just another kind of action a Step would take. All this audit information needs to be available to the executive recovery program. These identified open Steps will then be reviewed and worked through as the executive recovery program steps back through all of the audit trail data until no active processes (Steps, commits in progress, (or rollbacks if used)) that were found in the last recorded P-SAVE are discovered in what will be the final P-SAVE of the search to begin recovery. Actually, in the preferred embodiment in order to properly restore the database, an executive program will be preparing to do the next P-SAVE after each last one is stored. This next to be saved P-SAVE will contain the address of the last saved P-SAVE, so if the executive's memory is available, it will contain any Steps started after the last P-SAVE and the address of the last P-SAVE. Therefore, the actual record of open Steps and active processes that the system needs to look through will be found in the executive memory, or an audit trail it is writing to if it fails, and the last P-SAVE the location of which is found in the executive memory also. With reference to FIG. 4, the executive memory can be assumed to be managed by the Audit Info Record Manager 47 and be contained in the audit trail data memory 19, although any number of alternatives are available given the configuration of the system and software in which the invention will be used.

Detailed Description Text (16):
Please refer to FIG. 6, which illustrates a process 60 by which a user's account record can be modified in the transaction processing system. The first step in this process 61 has the user entering an input message with a transaction code to transfers funds from a savings account into a checking account. This would begin an exemplary step as the word step has been used previously in this document. In step 62 the database manager (manager 46 of FIG. 4) finds the program corresponding to the transaction desired by the user. The transaction program is loaded in to memory in step 63 and program read the input data from the user in step 64. In step 65 the program causes the system to update the transaction processing database so that the checking balance is increased and the savings balance decreased by the same amount. Sometime after this when the message is sent that the process has been complete in step 66, the end of the step is accomplished at point 72. Depending on the program, a rollback may require the initiation of a new transaction or may occur within step 66 if the program asks for confirmation from the user before step 65.

Detailed Description Text (21):
Thus, in FIG. 5C, step 108, the process will read the location stored for the previous P-SAVE in the current P-SAVE and the system can then be directed

immediately to the previous P-SAVE. Step 109 compares all the incomplete activities and step identifications that were located in the newly found P-SAVE that was just directly located to all the incomplete activities in steps in the executive recovery programs storage. Again, in step 109, steps that are no longer found to be open or incomplete are either deleted or marked. (Alternatively, and perhaps preferably, in none of the active Steps originally stored are found when a new P-SAVE is read, we can declare that this is the last P-SAVE needed). A determination is then made 52 of whether there are no remaining incomplete activities or steps or whether they have all been marked. If we have some remaining incomplete or open steps we return to step 108. If there are none remaining then we can proceed to step 53 allowing the executive recovery program to perform its function of recovering the data in the database that has been subjected to open or incomplete steps.

CLAIMS:

1. Method of reading an audit trail record prior to restoring data in a database system comprising: a. locating a last P-SAVE record on an audit trail, b. reading data in said last P-SAVE to determine which Steps or transactions are either incomplete or uncommitted, c. producing a retained record of all said incomplete or uncommitted Steps or transactions, d. reading the address of a previous P-SAVE from within data in said last P-SAVE, e. moving directly to the address of a previous P-SAVE from the last P-SAVE location, f. checking the information in said previous P-SAVE record to determine which of said all incomplete or uncommitted Steps or transactions in said retained record produced in step c are also incomplete or uncommitted in this previous P-SAVE, g. if there are no incomplete or uncommitted transactions or Steps in said previous P-SAVE, allowing restoring of data in said database to begin, else, iterate from step d until there are no incomplete or uncommitted transactions or Steps.

6. Method of storing information into an audit trail for use by a computer system which runs a database which contains data to be maintained by reference to said audit trail in event of failure in said computer system, said audit trail storing method comprising: continuously recording into an audit trail record, blocks of information desirable to have for a recovery of said database onto said audit trail, periodically recording an identification record of all incomplete Steps to a periodic save (P-SAVE) record in a location among said audit trail record blocks in said audit trail record, recording the location of said P-SAVE record within said audit trail record in a next P-SAVE block.

8. A system of audit trail record keeping for storing data on an audit trail for use with a large database having numerous database records to recover integrity of said large database's records when desirable to do so, wherein said database may be operated upon by user access to any said database records processor called Steps in which Steps employ a database manager operating on said database and wherein a system for storing step ID codes or names of active Steps is operating to maintain said database integrity, and wherein said Steps are opened upon a request for access to a database record and are closed upon completing of any action with respect to said request and wherein said system for storing step ID codes or names provides an indication of completed Steps, said system comprising: An audit information record manager for monitoring every user initiated Step affecting said database records in the database and for generating an audit record of each said Step, and for writing said audit record to an audit trail, said audit information record manager comprising a monitor path for monitoring said database manager wherein data from said monitoring path is used by said audit information record manager to generate a record for each said Step that requests access of any database record, via said database manager, Said audit information record manager further comprising a process for writing each said audit record into audit data blocks on an audit trail, an audit trail comprising a series of audit trail blocks written by said audit information record manager at a rate directly proportional to

how many Steps occur, and wherein a periodic save (P-SAVE) audit trail record is written to said audit trail at a predefined periodic rate, said periodic save audit trail record comprising all Step ID's codes or names that are not completed by said step table and processor during each said period, and said periodic save audit trail record further comprising an address within said audit trail of a prior P-SAVE audit trail record.

First Hit Fwd Refs

End of Result Set

☐ Generate Collection Print

L18: Entry 3 of 3 File: USPT Oct 5, 1999

DOCUMENT-IDENTIFIER: US 5963203 A
TITLE: Interactive video icon with designated viewing position

Detailed Description Text (26):
The designer first of all indicates for which video sequence he desires to create an interface, for example by typing in the name of a stored file containing the video sequence information. Preferably, the toolkit accesses this video sequence information for display in a window on the screen for consultation by the designer during the interface design process. In such preferred embodiments of the toolkit, the designer may make his selection of basic frames/objects for the root image, extractable objects and the like by stepping slowly through the video sequence and, for example, using a mouse to place a cursor on frames or portions of frames which are of interest. The toolkit logs the frame number (and x, y locations of regions in a frame, where appropriate) of the frames/frame portions indicated by the designer and associates this positional information with the appropriate parameter being defined. Preferably, at the end of the interface design process the designer is presented with a displayed view of the root image for manipulation so that he may determine whether any changes to the interface data file are required.

Detailed Description Text (60):
The data are then stored in a local memory unit (203) which may be either a cache memory, a disk store or any other writable storage element. The local memory unit (203) stores the 4 files created by the editor (see above) and in addition a transaction/audit file. In certain cases the applications programs are already resident in the interface viewer unit and so do not need to be transmitted.

Detailed Description Text (62):
The user interacts with the interface through the user interaction unit (205) to the navigation unit (206) and all his actions are audited by the transaction/audit unit (207). In addition, the user can interact with the transaction/audit unit (207) for example to supply any information required by the script which is then recorded and stored in the transaction/audit portion of the local memory unit (203). Depending upon the application, this transaction/audit file or a portion thereof is transmitted by the interface database manager to the appropriate storage unit (201). This information is then available for externally (optional) located auditing and transaction processing facilities/applications. In a typical situation, the auditing information is transmitted at the end of the session whereas the transaction information may be performed on-line, i.e. the transaction information is submitted during the session.

and continues storing them in multiple buffers, then writing them to the remote host 30 asynchronously until the Catch-up server 28 determines that the source audit trail in audit file 18 is synchronized with the receipt audit file 34 in the secondary host 30 after which it then stops (END) the process of transferring audit blocks.

Detailed Description Text (103):
It should be seen that a "partition" is illustrated in FIG. 9, sections 18a, 18b and 18c. "Section" is a logical splitting of audit files into segregated blocks. Further, the audit blocks which are received from the source are read asynchronously by the Catch-up task unit 31 in a fashion which bypasses any serial reading of the logical blocks but rather in a parallel fashion from the multiple Catch-up tasks which are utilized by tracker 36 for storage in the secondary database 38 in their logical order.

Detailed Description Text (104):
Thus, by the use of parallel transfer modes for transferring the logically partitioned audit files, there is a more efficient transfer of audit blocks at a higher speed in order to expedite the synchronization process.

Detailed Description Text (105):
Referring to FIG. 9, there is shown a schematic diagram which illustrates the parallel and concurrent transfer of sectioned audit files from the primary host to the secondary host or remote host.

Detailed Description Text (106):
The remote Database Support Library 35 communicates with the Catch-up Server Task program 28 which, for illustrated purposes, is seen to have several sectioned audit files designated 18a, 18b, and 18c.

Detailed Description Text (107):
Then by use of the Catch-up port file 10c, 30c, FIG. 3 (CU.sub.13 PORT port file), these sectioned audit files (18a, 18b, 18c,) are transferred over to the secondary host audit file and indicated as audit files 34a, 34b, and 34c.

Detailed Description Text (108):
Thus, in cooperation with the first Remote Database Support Library 20 and the secondary Remote Database Library 35, the Catch-up Server 28 can now transfer the sectioned audit files as multiple sectioned groups to be attended to by a series of catch-up process operations (31a, 31b, 31c) which work together in parallel to asynchronously transfer the sectioned audit files to the Tracker 36 in the secondary host for placement onto the secondary database 38.

Detailed Description Text (109):
While earlier systems used only a single serial transfer mode for audit block conveyance to the secondary host, the present system now enables multiple sectioned audit files to be simultaneously and concurrently transferred to the secondary host for placement on the secondary database 38.

Detailed Description Text (110):
As a result, the secondary (backup) database 38 is maintained in useful synchronism with the primary (source) database 14. That is to say, that now the secondary database accurately reflects the data in the primary database at any given time period.

Detailed Description Text (111):
Described herein has been a Tracker mechanism which enhances the method and system for synchronization of audit trails of a primary source host and remote secondary host to enable a secondary database to serve as backup to a primary database. The Tracker program senses the level of synchronization between the audit files of the

primary host and secondary host to determine if there is sufficient lag (out of synchronization) between primary and secondary audit trails. If this is the case, a Catch-up program is then invoked which divides the audit into sections which are expeditiously transferred in a parallel asynchronous basis from primary to secondary host for placement on a secondary backup database.

CLAIMS:

1. In a network wherein a primary host is connected to a secondary host in order to set up a secondary host backup database which will maintain data file synchronism with a primary host database, a method for sensing the level of duplication between said primary and secondary databases comprising the steps of: (a) sensing when audit blocks of sectioned audit files at said secondary host are not on a par with the number of sectioned audit block files residing at said primary host; (b) initiating a Catchup program to speed up the transfer of said audit blocks of sectioned audit files from primary host to secondary host in an asynchronous transfer fashion when the number of audit blocks at said secondary host falls below the number of audit blocks waiting for transfer from said primary host.

2. The method of claim 1 wherein step (a) includes the steps of: (a1) recognizing an End-Of-File condition after transfer of audit blocks of sectioned audit files from said primary host to said secondary host; (a2) scanning the audit block serial numbers of the audit blocks assigned by said primary host and transmitted to said secondary host; (a3) accessing data from a database support library to indicate any disparity between the audit block serial numbers, assigned in the primary host, to the audit block serial numbers received at said secondary host.

8. In a system for expediting the transfer of audit files from a primary host to a secondary host, a method for sensing the state of synchronism between sectioned audit files in said primary host awaiting transfer and sectioned audit files positioned as audit blocks in said secondary host which are received from said primary host, said method comprising the steps of: (a) sectioning said audit files into separate physical audit files designated as audit blocks and assigning a separate identification number to each audit block at said primary host; (b) placing each audit block in a separate sectioned based buffer at said primary host; (c) transferring each audit block at said primary host as permitted by the condition of the network connection between said primary and secondary host and establishing a special port connection between said primary and secondary host; (d) placing each audit block received at said secondary host into a separate sectioned based buffer; (e) sensing the number of audit blocks at said primary host which have not yet been transferred to said secondary host; (f) initiating a transfer speed-up program at said secondary host when the audit blocks received at said secondary host is a pre-set number less than the number of sectioned audit blocks ready to be transferred from said primary host.

9. In a system utilizing an audit block write mode method for transfer of audit file data from a primary host to a secondary host, a method for sensing any delay in the transfer of audit block sections from said primary host to said secondary host comprising the steps of: (a) placing an audit block serial number on each audit block in said primary which is destined for transfer to said secondary host; (b) transferring said primary host audit blocks over to said secondary host as permitted by the operating condition of interconnecting network communication lines; (c) calculating the difference value between the highest audit block serial number received at said secondary host and the highest numbered audit block serial number written in said primary host.

12. A method for detecting the condition of out-of-synchronization between sectioned audit files of a source host database and a remote host database, comprising the steps of: (a) assigning an identifying serial number to each audit block in a sectioned audit file at said source host; (b) checking periodically to

compare the highest contiguously written audit block serial number of said source audit blocks with the highest contiguously written audit block serial number of said remote audit blocks; (c) initiating an expedited transfer of audit blocks from said source database host to said remote database host when the said highest source audit block serial number is greater than said highest remote audit block serial number.

13. A system for re-instituting the transfer of audit blocks after an interrupted or failed network transmission connection between a primary host and a secondary host, and where each audit file block has been assigned a special serial number, said system comprising: (a) means to determine the highest audit block serial number, H, at said secondary host which has been received in said secondary host; (b) means to store the audit block serial numbers at said primary host in order to select the serial number, H+1, for initiating a sequence of audit block transmissions from said primary host to said secondary host.

L16: Entry 7 of 12 File: USPT Jun 25, 2002

DOCUMENT-IDENTIFIER: US 6411969 B1
TITLE: Enhanced system and method for management of system database utilities

Abstract Text (1):
An enhanced method for developing back-up copies of a source database by providing incremental and accumulate dump commands from various multiple-Users which enable a selection of certain files which are identified independently of time-factor for dumping selectively either onto a separate destination medium of disk or tape. A User can determine the block size of words for blocks of data files to be dumped onto the destination medium, thus significantly reducing the number of I/O operations required.

Parent Case Text (2):
This application is related to a co-pending Application, U.S. Ser. No. 09/322,535 entitled "Multi-Processor System for Data Base Management", and to a co-pending application, U.S. Ser. No. 374,352 entitled "Database Backup System and Method Utilizing Numerically Identified Files for Incremental Dumping" which are incorporated herein by reference.

Brief Summary Text (2):
This disclosure relates to improvements in the field of database back-up to disk and tape.

Brief Summary Text (7):
Another way is (ii) designated logical level back-up. This refers to the data as seen by the user application programs in files or in database tables. Normally, the operating system of the computer will include a file system that does mapping between the physical level and the logical level. On doing the physical level back-up, this would involve making a raw copy from a computer disk to some other storage medium without going through the file system or some other physical to logical interpreter module. Then on the other hand, the back-up using the logical level would involve using such a thing as an interpreter module or some sort of a file system while doing back-up of physical to logical mapping.

Brief Summary Text (8):
A Database management System consists of a set of tools used to develop and manage a database. The present system utilizes a DMSII which is a Database Management System available on the Unisys Corporation's ClearPath HMP NX, and the Unisys A-Series systems. A background for the Unisys DMSII systems is available in a publication of the Unisys Corporation, Document 8807 6625 000, entitled "Getting Started With DMSII" and published in September, 1997 by the Unisys Corporation. The DMSII Utilities provide database back-up and recovery capability for the entire database or for partial databases. The background operations of the DMSII utility enhancements are published in a Unisys Corporation publication Document 98037/4 and entitled "DMSII Utility Enhancements" published on Mar. 31, 1999.

Brief Summary Text (9):
Database back-ups can be accomplished for "on-line" and "off-line" bases. The on-line back-up will allow users to update data in the database, whereas the off-line back-up disallows all updates to the database. The back-ups can be done to either

tapes or disks or any combination of both types of such media.

Brief Summary Text (10):
Database Management Systems are used by many large and small businesses such as airline reservation systems, financial institutions, retail chains, insurance companies, utility companies and government agencies. The present Database Management System (DMS) in its form as DMSII is used to build database structures for items of data according to some appropriate logical model, such as relational, hierarchical, or network. Further, the Database Management System is used to manage the database structures and keep the structures in some stable order while various application programs may be retrieving or changing the data. The present embodiment of DMSII has a data definition language designated as Data And Structure Definition Language (DASDL).

Brief Summary Text (11):
There are various tasks that are performed in database management and these involve (i) monitoring and optimizing database performance; (ii) the use of database control for monitoring multi-program database access; (iii) the function of the data integrity and safety done by integrity checking and preventing access to the same data by multiple applications occurring at the same time; (iv) the function of defining data structures and the data fields within them, including the function of modifying data structures; (v) data access operations and developing an application program to retrieve data or to change data; (vi) the function of data shareability to provide multi-program access without conflicts and provide database definitions to the application program; (vii) in database and data security, to prevent unauthorized database access; (viii) ensuring independence of application programs from certain data changes and preventing the revision of application programs every time a structure changes; (ix) in database and data recovery, performing the resumption of database operations after an interruption; (x) tracking data changes by keeping a record of every change made to the data; (xi) for data change integrity, ensuring that update changes are applied to, or removed from, the database in their entirety; (xii) providing a recent copy of the database as a reserve by backing-up the database and storing copies of audit files and all other database files; (xiii) providing for database scaleability by growing or shrinking the database according to the ongoing needs at the time.

Brief Summary Text (12):
The DMSII provides standard software files that perform services and operations for all the databases connected to the system's Enterprise Server. This enables a viewing of a list of all these files on the user terminal.

Brief Summary Text (13):
In the ordinary course of operations, the application program user will submit changes to data or retrieve data while running a particular application program. Then, changes can be made which add, modify and delete data. A Database Administrator (DBA) keeps the database running smoothly and enforces the rules for data integrity and security. Users access the database through a given application program which itself does not access the data directly. Instead, the program interacts with the DMSII software and the database tailored software, which is directed by the access routines of the Data Management System to provide accesses, retrievals and the storage of data in the physical database file.

Brief Summary Text (14):
In regard to access, an application user will access the data in order to (i) make an inquiry to get a Read of data in the database, or (ii) to provide an update by doing a Write to the database thus, adding, deleting or changing data. The access for either purpose contributes to an operation on the database which is called a "transaction".

Brief Summary Text (15):

A transaction is a sequence of operations grouped by a user program because the operations constitute a single logical change to the database, At the end and finality of the transaction point, the transaction is complete and without error, and it is considered as being committed to the database.

Brief Summary Text (16):
Actual real world data goes into special logical structures that are used by the Data Management System to store data. The database is designed to map categories of data into suitable structures. For example, the real world data would have a character with, structure called a "data set". An example of this would be a particular person's name. Then, real world data that can serve as an index of a whole data set has a structured name called a "set". This, for example, might be the social security number of any employee. Then there is data that can serve as an index of a data set under a certain condition, and this is called a "subset". This might be an employee's work number, for example. Then, there is data about each instance of a particular category and the structure name for this is "data item". An example of this might be the name and address of the category (person). Then, there is data related to the database as a whole, and this involves a structure called "global data item". An example of this might be the total number of employees in a company. Once there has been identification of the real-world data which is to be stored in the database, it is then necessary to define that data in relationship to the data structures of the data management system that holds data. When this data is defined within "structures", then the data management system and the system software programs an application program that can then understand how to make this data accessible for various inquiries and/or changes. This is done with the Data and Structure Definition Language (DASDL).

Brief Summary Text (17):
The Data Management System structures are the building blocks of the Data Management System database. Here, the "data set" has the purpose of storing data pertaining to a data category in a collection of records. A "set" has the purpose of indexing all records in a data set. A "subset" serves the purpose to index some records in a data set according to some given criteria. The "data item" is a structured name which defines a unit of information about a category in a given field (column) of a data set record. A "global data item" serves the purpose of storing a unit of information about the entire database or any of its involved structures. In general discussion about the types of data and the names of data structures, it is often seen that in a relational database, a "data set" is called a "table". A "set" or "subset" is frequently called an "index". A "data item" is often called a "field" or a "column", or is often called by its data name, for example, a project number. "Structures" are made of common file components designated as records and fields.

Brief Summary Text (23):
A data set is kept up-to-date in several ways: (i) here, application programs add, change, or delete individual pieces of data or records stored in the data set; (ii) the Database Administrator (DBA) maintains the structure of the data set by keeping the data set within certain maximized limits, by adding, deleting or changing the definition of a data item, creating new sets or subsets, monitoring automatic processes that guard data integrity and creating guard files to enhance the security of the data.

Brief Summary Text (26):
For example, an application program may compile a list of people who are "managers". Thus, it is seen that the database designer created the "manager" subset. Thus, in order to retrieve a record of managers, the data management system can use the smaller file, that is, the subset, to quickly point to the corresponding records in the larger file which is the data set. As with the set, the subset must also be kept up-to-date.

Brief Summary Text (27):
A "data item" is an element of data. In the Data Management System, a data item can also be the field (column) in the database record. For example, the social security number could be considered as a data item in the sample data set designated "person". The purpose of the data item is to describe the data to be stored. The data item provides the identity - - - type, size, location, and attributes - - - of one element of data for a database entity when an application submits an update to a data item, the Data Management System will accept the update if it corresponds to the definition of a data item. Otherwise, the change is rejected and reported as an exception. The Database Administrator will add, delete or change the data item definitions. There are a number of data items that are used by the Data Management System. These include the type called "alpha-numeric" which include words and characters, names, addresses, dates and titles. Then, there are data items designated as "numeric" which involve integers and decimals with or without signs. Then, there are data items designated as "real" which involve single precision floating point numbers that occupy one word. An example of this would be, for example, an employee's salary. Then, there are data items which are called "Boolean" which involve TRUE and FALSE values.

Brief Summary Text (28):
The "global data item" is a data item, a group item, or a population item that is not part of any data set but still pertains to the database as a whole. Such global data items are stored in one special record called the "global record" in the DASDL declaration which is outside the structured definitions. Sometimes the global record is placed just before the structured definitions in the DASDL file. The global data item has the purpose of holding permanent information about the database as a whole or about a particular data set. It also acts as a place holder for information that can be derived from the database.

Brief Summary Text (29):
One of the most significant options in DASDL (Data And Structure Definition Language) is that it is possible to define the database as to whether the database is to be audited. The data management system supports both logging changes to a database (auditing the database) or not logging changes (maintaining an unaudited database). There are advantages in auditing a database since this assures the user that if a database failure occurs, there will be a record of database changes with which one can restore the database to a completely integral state and thus avoid loss of information and corruption of information.

Brief Summary Text (30):
The "audit trail" is a log of changes made to the database. This type of audit trail is somewhat similar to the SUMLOG in the host system which is history of all system activity except for the fact that the audit trail will record the database update activity only and will consist of separate numbered files. Thus the data management system software can use an audit trail to recover the database from an unusable state, provide restart information to user programs, reconstruct portions of the database that had been lost because of hardware errors, back out aborted transactions and roll back the entire database to a user specified point or rebuild the entire database to a user-specified point.

Brief Summary Text (31):
The "audit file" provides a chronological history of all update database transactions. The audit file is a numbered segment of the database audit trail where the data management system assigns each audit file to have an audit file number (AFN) in the range of 1 to 9999.

Brief Summary Text (32):
Access Routines Program: The data management system controls access to database data with a software program called Access Routines which is a collection of specialized routines that enables many users to access the database all at the same

time and ensures that the access is controlled so that accesses do not conflict with one another.

Brief Summary Text (33):
Control File: Each active data management system database has a control file. The control file contains the time stamps for the database software and files and the access routines since the access routines use time stamps to check the validity of data. A control file also contains the update levels of the database and the structures since the access routines use update levels to check the validity of data. Further, the control file functions to store audit control information, dynamic database parameters plus other information. It further guards the database from interruption while a process that needs exclusive access to the database goes on to complete its task successfully, such as, for example, a halt/load recovery and/or a reorganization. The control file assures that a database that has been interrupted for any reason is not accessed until the integrity of the database is further guaranteed by the successful completion of the recovery process.

Brief Summary Text (35):
Backup: The most important preventive maintenance task which can be performed for a database is to back up the database frequently and to keep the backups for some period of time. To "back up" the database, means to use the data management system DMUTILITY program to make a copy of all or part of the database. This backup will include a check of the physical integrity of all the database's structures being backed up. A complete database includes providing a reserve copy of all the files pertaining to the database. All the files include not only the database files and the control files (which may change from time to time) but also the DASDL source file, the description file, various tailored files, application programs, and audit files. This enables a user to put the database back in operation quickly in case the current database files should become unavailable or damaged.

Brief Summary Text (36):
Here there is involved the concept of "DUMP." A DUMP involves either a copy of stored data in which a change has been made since the previous DUMP of that data or a transfer of all or part of the contents of one section of computer storage to another section or to some other output device. The processes used to make a database are called "backing up" and "Dumping." A backup to tape is called a "Tape DUMP" while a backup to disk is called a "Disk DUMP."

Brief Summary Text (37):
Often the backing up operation for the database is done by increments. An increment is one of the series regular consecutive additions, for example if a database is too large to back up on a daily basis, the operator could create a schedule that backed up a certain number of database files (an increment) each day until the entire database was backed up.

Brief Summary Text (38):
The dump of a database is done to tape or disk depending on what type of storage resources are available. Tapes are most frequently used since they are the less expensive resource than disk. When dumping is done to tape, it is necessary to furnish information common to any disk-to-tape process and this information would include the tape name, the cycle number, the version number, workers, the serial number, compression and non-compression, the density, and the SCRATCHPOOL option.

Brief Summary Text (40):
Recovering a database means to get it back and ready up to date, ready for access with complete and correct data. The recovery of the database can be done either automatically or be done manually using various software utilities and commands.

Brief Summary Text (41):
The present system and method provides enhancements which accomplish performance

improvements in the DMSII database utilities, plus support for new tape devices and other efficient back-up methods. The presently-described system achieves special optimization of Input and Output in both disk or tape operations, plus special DUMP features which enhance the ability to perform database back-up. As a result, total back-up time has been reduced due to the ability to DUMP the data blocks which have been modified since the last DUMP. Thus, the presently described system will provide users with more options for improving the efficiency of their database administration and operational practices.

Brief Summary Text (43):
A system and method is provided to enhance the ability to perform database back-up using special features designated as the "Incremental DUMP" and also a feature designated as "Accumulated DUMP". As a result, the total back-up time has been considerably reduced due to the ability to dump the data blocks which have been modified since the last DUMP. This enables the data recovery process to operate more efficiently due to the lesser number of audit images that are being applied.

Drawing Description Text (6):
FIG. 5 is a flow chart for providing back-up from a database via a DUMP process using the selected block size as a physical block size in words;

Drawing Description Text (8):
FIG. 7 is a flow chart involving the providing of a back-up from the database for dumping to a tape using the selected block size for this operation.

Detailed Description Text (22):
COMPARE PHASE: After each database backup session, the DMUTILITY program automatically issues the compare operations to verify that the data written to tape or disk is free from block checksum, block sequencing, and I/O errors. This phase is referred to as the compare phase of the backup operation.

Detailed Description Text (25):
DASDL: Data And Structure Definition Language for defining the database.

Detailed Description Text (26):
DATA MANAGEMENT SYSTEM II (DMSII): A specialized system software package used to describe a database and maintain the relationships among the data elements in the database. Described in Unisys Publication Part No. 8807 6625-000, September 1997, entitled "Unisys: Getting Started With DMSII".

Detailed Description Text (27):
DBA: The definition, design, maintenance, and use of a database (DMSII or other databases) and its related products which should be coordinated and controlled by a single administration. This control is typically established in the function of a Data Base Administration, abbreviated as DBA.

Detailed Description Text (31):
DMDUMPDIR: A program that retrieves and modifies directory information for database dumps. When this program is enabled, directory entries are created automatically when a new database dump is created, or existing database dump is copied, or existing database dump is duplicated.

Detailed Description Text (36):
DUMP TO TAPE SESSION: A database backup is a snapshot of an entire database or of parts of a database. The backup can later be used to recover lost data, or transfer the database from one location to another. One of the features in a DMUTILITY DUMP command is to create a backup copy of the database on tape. When this command has been executed, a Dump to Tape session will be initiated.

Detailed Description Text (38):

EXCLUDE KEYWORD: New syntax to be added to DMUTILITY program to allow users to exclude certain database files in a dump command. This is useful when the excluded files have been backed up previously and no new changes have been made that require new backup to tape or disk.

Detailed Description Text (39):
EXCLUDE PARAMETER: This is for the DMUTILITY Dump command which has the purpose of excluding one or more structures from the database dump. This new feature provides flexibility to the current Dump command and is especially helpful when a small percent of the structures are being excluded from the Dump operation. This command is supported for the DMUTILITY Dumps with all the files selected, and it allows user to select one or more disjoint data sets and all of the structures associated with it (sets, subsets, embedded structures, etc.) which are to be excluded from each dump session.

Detailed Description Text (67):
SERIAL NUMBER: This involves the display serial number whereby the program DMUTILITY will display the serial number of the tape, which contains the latest directory information in the dump output listing. This information is helpful for Database Administrators for subsequent DMUTILITY runs. In old systems, the Serial Number information was displayed when DMDUMPDIR is enabled. However, this can be improved in order to have this available in the DMUTILITY program without the need to use DMDUMPDIR.

Detailed Description Text (68):
SERIAL NUMBER INTERFACE: This is for the purpose of displaying the tape serial number of the latest version tape in the dump output after each Dump to Tape session has completed. The tape serial number information is currently available by enabling the option DMDUMPDIR. In most cases, a database fits in multiple volumes and there will be only one serial number that the DBA needs to know, for obtaining the latest version of tape, and this information will be available through DMUTILITY.

Detailed Description Text (77):
The present system involves a more rapid and efficient method of functionality for the back-up of a database. As a very important preventive maintenance task, it is essential that the database be backed-up frequently and kept for an extended period of time. Backing-up the database means, in this situation, to use the DMSII DMUTILITY program in order to make a copy of all or part of the database. The back-up operation will include a check of the physical integrity of all of the database structures being backed-up.

Detailed Description Text (78):
It is generally recommended that a back-up be done every day and possibly even more often if there are special circumstances involved. One typical operation is that of making changes to the database structures as a special operation and during which such operation backups should be utilized both before the changes and after changes. A complete database back-up includes a reserved copy of all of the files pertaining to the particular database. All these files include not only the database files and the control file, but also the DASDL source file, the description file, the tailored files, the application programs, and the audit files. Thus, by having a reserved copy of all the files necessary for the database, this enables the Users to put the database back into operation quickly should any situation occur where the current database files become damaged or unavailable or lost in some fashion.

Detailed Description Text (79):
An essential part of the back-up procedure for database files is the use of a Dump. The Dump refers to either a copy of stored data in which a change has been made since the previous Dump of that data, or the transfer of all or parts of one

section of the computer storage to another section, or to some other output device.

Detailed Description Text (80):
Using the DMUTILITY program in a database back-up, the processes which are used to make a database back-up are called backing-up and/or Dumping. The backed-up database is called a back-up or a Dump, while a back-up to a tape peripheral is called a Tape Dump and a back-up to a disk peripheral is called a Disk Dump.

Detailed Description Text (81):
For further security, it is often recommended to make a back-up of the back-up file. The reasons for this is that files can be deleted or made unusable so that it is possible to use certain of the following commands to back-up the database dump. These are:

Detailed Description Text (84):
The Dump command is used as part of the database back-up. The files that the Dump command backs-up are either some or all of the database files and the control file. There are several types of management utilization in backing-up these files so that the User may want to back-up the files only when they change, or else to use utilities other than the DUMP command to copy the files.

Detailed Description Text (85):
In order to back-up the database files other than using the DUMPS and to have all database files backed-up in case the User needs to re-establish the current database from scratch, there are certain files that are required to be held in reserve and these are:

Detailed Description Text (87):
(b) The Audit files. Copies of both the primary and secondary audit trails;

Detailed Description Text (89):
(d) The Database Description file.

Detailed Description Text (90):
Thus, the User would back-up a database application program after a database reorganization, for example, that closes the application program to further changes.

Detailed Description Text (92):
Two dump commands, Incremental and Accumulated, are introduced to allow the Users the option to dump only those portions of the database which have changed since the last dump. These dumps are generated based on the last update time of each data block for each structure in the database. These features can potentially reduce the total time required to perform database dumps. For incremental dump, all data blocks which have changed since the last full, incremental or accumulated dump, are backed-up. For accumulated dump, all data blocks, which have changed since the last full dump, are included in the dump.

Detailed Description Text (96):
At step (B), the Operating System of the MCP is connected to the Data Management System II. The Data Management System (DMSII), as previously noted, is a specialized system software package used to describe a database and to maintain the relationships among the data elements in the database. Then, the sequence proceeds to step (C) which invokes the DMUTILITY program which will parse the newly-provided syntax in order to scan for a keyword designated as the "EXCLUDE" keyword, thus, to build a DUMPLIST TO reflect the specific disjoint data set and all of its sub-level structures which are to be excluded from the DUMP.

Detailed Description Text (99):

Now returning to step (C) and on to the branch (CT) which involves a back-up operation to tape. From step (CT), the next sequence is step (ET), which is an execution of a Dump-to-Tape operation. Since the database back-up is actually a snapshot of the entire database or merely parts of the database, it is essential to have a back-up which can later be used to recover lost data or even to transfer the database from one location to another. Thus at step (ET), a Dump-to-Tape session is initiated, where at step (G1) there can be a Read operation from a source database 40s and also a Write operation at step (G2) which provides the destination database to placed on tape 50.

Detailed Description Text (112):
The reference marker DA is shown to indicate that the sequence proceeds to FIG. 5. In FIG. 5 from the reference mark DA, the sequence proceeds to step (ED7) which involves getting the next file in the database, after which the next step (ED8) involves a decision block to query if there is any more data in this file. If the answer is (YES), then the sequence proceeds to step (ED9) which involves reading the next group of contiguous sectors from the source file, such as 40S in FIG. 2.

Detailed Description Text (118):
At step (ET7), the sequence will get the next file in the database and then proceed to step (ET8) which involves a decision block to query whether there is any more data in this particular file. If the answer is (YES), the sequence proceeds to step (ET9) where there will be reading of the next group of contiguous sectors taken from the source file which might be, as shown in FIG. 2, at the source data base 40S.

Detailed Description Text (122):
It may be noted that the DMUTILITY does not actually issue the positioning commands to the tape drive, but if the write operations are not done quickly enough or if not enough data is written out to the tape, there is always the possibility that the tape device (at the hardware level) will lose its position. When the software program issues the next Write, then the tape drive will need to reposition itself to the place on the tape adjacent to the last Write operation. Since DMUTILITY does not have control over the repositioning operation performed by the tape device, there is still the necessity to ensure that the latency or repositioning lag does not hold-up the data transfers because this will impact the performance of the database back-up and recovery.

Detailed Description Text (124):
Described herein has been an improved system for producing back-ups for a source database. While one embodiment of the invention has been illustrated there may be other embodiments which are still encompassed by the subject invention as described by the following claims.

CLAIMS:

1. A system for enabling multi-Users to operate concurrently for enhancing the function of backing-up data files from a source database means have multiple sources to a storage destination medium means having multiple receiving media comprising:

(a) utility program mans concurrently usable by multiple-Users for copying data files from said source database means by transferring data files via expandable buffer means to a series of data blocks;

(b) means for determining whether the data files are to be destined for a disk storage medium or a tape storage medium);

(c) means for placing (DUMP) said copied data files onto said disk means or tape storage medium means, said means including:

(c1) means for initiating a DUMP to Disk or DUMP to Tape command as selected by a participating User,

(c2) means for reading blocks of data files from said source database means a writing said data files onto a destination medium means of multiple Disk or Tape units, said means including;

(c2a) means for selecting the block size by a participating user of selected data files by determining the number of words to be placed in each data block;

(c2b) means for placing the newly-determined-size blocks of data files onto said destination medium.

4. In a computer system having multiple processors each operating under a Master Control Program with a main memory and I/O module connected to a source database means, a method for instituting a dump for developing a back-up copy of data from said source database means comprising the steps of:

(a) utilizing a utility program available to multiple Users to initiate the copying of data files from said source database means to a destination medium means;

(b) altering the data block size of the destination dump files by selective choice of a participating User for better compatibility of storage on said destination medium means and for reducing the number of I/O's required;

(c) placing said copied data files in their newly-formed block format onto said destination medium means.